Semarchy Convergence for MDM

Getting Started Guide

3.0, Revision 1

Getting Started Guide

Overview

Welcome to **Semarchy Convergence for MDM**!
This Getting Started tutorial provides a step-by-step introduction to **Semarchy Convergence for MDM** (Master Data Management).

> **Tip**: If you want to learn about MDM or discover Semarchy Convergence, please visit the Semarchy Demo Center.
>
> **Note**: The **Semarchy Convergence Documentation Library**, including the development, administration and installation guides is available online at the following URL: http://semarchy.com/download/

Contents

Audience

This document is intended for users interested in learning how to use the **Semarchy Convergence for MDM** for their Enterprise Master Data Management Initiatives.

Document Conventions

This guide uses the following formatting conventions:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or a product specific term or concept.
italic	Italic type indicates special emphasis or placeholder variable that you need to provide.
`monospace`	Monospace type indicates code example, text or commands that you enter.

Other Semarchy Resources

In addition to the product manuals, Semarchy provides other resources available on its web site: http://www.semarchy.com.

Obtaining Help

There are many ways to access Semarchy Technical Support. You can call or email our global Technical Support Center (support@semarchy.com). For more information, see http://www.semarchy.com/.

Feedback

We welcome your comments and suggestions on the quality and usefulness of this documentation.
If you find any error or have any suggestion for improvement, please mail support@semarchy.com and indicate the title of the documentation along with the chapter, section, and page number, if available.
Please let us know if you want a reply.

Overview

This Getting Started tutorial provides a step-by-step introduction to Convergence for MDM (Master Data Management).

What is Convergence for MDM?

Semarchy Convergence for MDM is designed to support any kind of Enterprise Master Data Management initiative. It brings an extreme flexibility for defining and implementing master data models and releasing them to production. Convergence for MDM can be used as the target deployment point for all master data of your enterprise or in conjunction with existing data hubs to contribute to data transparency and quality with federated governance processes. Its powerful and intuitive environment covers all common use cases for setting up a successful master data governance strategy.

For more information, see:

- Why Do I need MDM?
- What is Evolutionary MDM?
- Semarchy Convergence for MDM Product Tour

Tutorial Overview

In this tutorial, you will:

- **Design** a master data hub containing employees, customers and cost centers information. You will also configure the **Integration Rules** to load, enrich and consolidate data coming from several distinct source systems and produce certified golden records.
- **Deploy** the MDM Hub.
- **Integrate** data from various source systems into the hub, and review the consolidated data.
- **Design an Application** for business users and data stewards to access the MDM hub data.
- Use the application to **Browse** the master data and modify this data through **Human Workflows**.

Installing Semarchy

Requirements

Before starting this tutorial, make sure that the following requirements are met:

1. Oracle Database version 10.2 or above is installed and configured.
2. A Java Runtime Environment (JRE) or Development Kit (JDK) 1.6 (Update 24 or above) is installed and the JAVA_HOME or JRE_HOME environment variable is configured to point to this installation of Java. Set JAVA_HOME to your JDK installation directory (e.g., “c:\Progra~1\java\jdk1.6.0”) or set JRE_HOME to the JRE base directory (e.g., “c:\Progra~1\java\jre1.6.0”).

Getting the Oracle Database

Oracle Database can be downloaded for free for Linux and Windows at the following URL: http://www.oracle.com/technetwork/database/enterprise-edition/downloads/index-092322.html

> **Note:** Convergence for MDM works with Oracle Express Edition (not available for Windows 64bits machines). It also is possible to use with any other edition of Oracle (Standard or Enterprise). These are free to use for the purpose of developing, testing and prototyping. Note also that Amazon Web Services offers Oracle as part of Cloud Relational Database Service.

If you have any difficulty figuring out a good solution for getting Oracle, please contact support@semarchy.com

Installing the Semarchy Convergence for MDM Demo

Configure the Database Schemas

Convergence for MDM uses three schemas for the demonstration environment:

1. SEMARCHY_DEMO_REPOSITORY contains the Convergence for MDM Repository.
2. SEMARCHY_DEMO_MDM is the schema into which you will deploy your MDM Hub.
3. SEMARCHY_DEMO_SOURCE contains sample source data used to load the MDM Hub.

To configure the database schemas:

1. Connect with a system account to the Oracle Database.
2. Run the following script to create the Convergence for MDM demo schemas:

```
CREATE USER SEMARCHY_DEMO_REPOSITORY IDENTIFIED BY
SEMARCHY_DEMO_REPOSITORY DEFAULT TABLESPACE USERS
TEMPORARY TABLESPACE TEMP;

CREATE USER SEMARCHY_DEMO_SOURCE IDENTIFIED BY
SEMARCHY_DEMO_SOURCE DEFAULT TABLESPACE USERS
```

```
TEMPORARY TABLESPACE TEMP;

CREATE USER SEMARCHY_DEMO_MDM IDENTIFIED BY
SEMARCHY_DEMO_MDM DEFAULT TABLESPACE USERS
TEMPORARY TABLESPACE TEMP;

GRANT CONNECT,RESOURCE TO SEMARCHY_DEMO_REPOSITORY,
SEMARCHY_DEMO_SOURCE, SEMARCHY_DEMO_MDM;
```

If you have already created these schemas and wish to delete them beforehand, use the following script:

```
DROP USER SEMARCHY_DEMO_REPOSITORY CASCADE;
DROP USER SEMARCHY_DEMO_SOURCE CASCADE;
DROP USER SEMARCHY_DEMO_MDM CASCADE;
```

Install and Start the Convergence for MDM Server

Convergence for MDM runs as a web application in a Java Application Server. A simple Apache Tomcat Server, pre-configured with Convergence for MDM is used for this tutorial.

In the following section, the `semarchy_with_tomcat.zip` file refers to the *Semarchy Convergence for MDM – Full Setup with Apache Tomcat* file that you can download to install Semarchy Convergence for MDM. The name of this file varies as it includes the Convergence for MDM version and build number.

1. Download the Semarchy Convergence for MDM distribution. Make sure to download the version that includes the pre-configured Tomcat Server. It is named `semarchy_with_tomcat.zip`.
2. Install and start the Apache Tomcat Server:
   - On a Windows Platform:
     1. Uncompress the `semarchy_with_tomcat.zip` file in your machine. This will create a `\semarchy` folder.
     2. Open Windows Explorer, and go to the `\semarchy\bin`.
     3. Run `startup.bat`.
   - On a UNIX/Linux Platform:
     1. Uncompress the `semarchy_with_tomcat.zip` file in your home folder. This will create a `$HOME/semarchy` folder.
     2. In a Shell window, run `$HOME/semarchy/bin/startup.sh`.

   Warning: The Apache Tomcat server starts by default on the port 8088. If this port is already used by another application and you wish to start the server on a different port, edit the `/conf/server.xml` file with a text editor, and change the port value in the following line: `<Connector port="8088" protocol="HTTP/1.1"`

   Warning The Convergence for MDM server is preconfigured with three JDBC datasources pointing to the three schemas you created (called `SEMARCHY_DEMO_REPOSITORY`, `SEMARCHY_DEMO_SOURCE` and `SEMARCHY_DEMO_MDM`), stored in an Oracle XE instance started on the local machine and listening on port 1521 (that is: `localhost:xe:1521`). If the Oracle instance that you are using is configured differently: First, stop the Convergence for MDM server,

edit the `/conf/catalina/localhost/semarchy.xml` to change the configuration of the datasources and then restart the Convergence for MDM server.

To shut down the Convergence for MDM server, run the `shutdown.bat` (Windows) or `shutdown.sh` (UNIX/Linux) script in the `/bin` folder.

Install the Repository

Convergence for MDM holds all its information in a repository stored in a database schema. The first task when connecting Convergence for MDM is to create this repository structure in the database schema previously created.

1. Open your web browser and connect to the following URL: `http://localhost:8088/semarchy/workbench/` (update the port number if you changed it during the installation process)
2. In the login prompt, enter the following:
   - **User**: *semadmin*
   - **Password**: *semadmin*
3. The Convergence for MDM Workbench opens with the license agreement. Review the End-User License Agreement.
4. Check the **I have read and accept Semarchy's End-User License Agreement** box and then click **Next**.
5. In the **License Key File** page, select a valid license key by clicking the **Upload License Key file...** button and then click **Next**.
6. In the **Repository Creation** wizard, select **Design** for the type of repository and then click **Finish**.
7. Click **OK** when the *Repository Successfully Created* message appears.

The repository has been created and Convergence for MDM is now up and running.

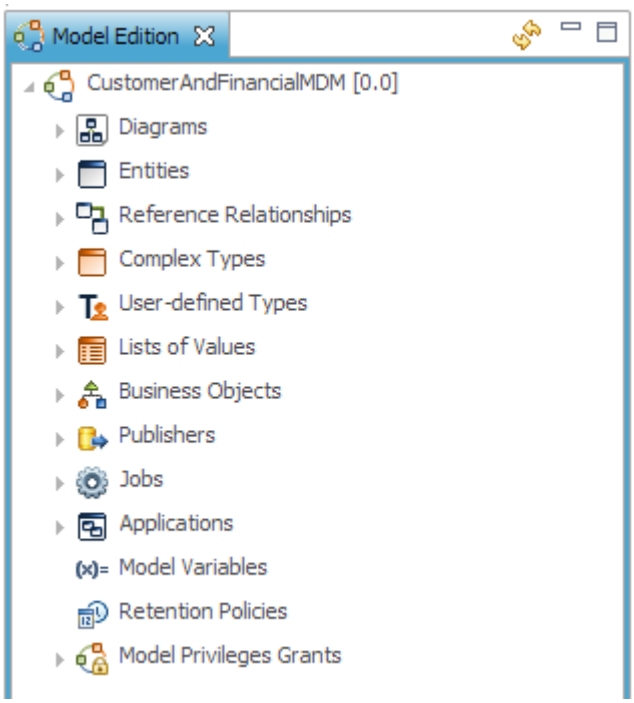
Create the Demo Model

The Tutorial Environment contains a *Customer and Financial Hub* model. This model simulates an MDM project in progress. During this tutorial, you will finish the design of this model, deploy the MDM Hub and load it from sample data sources.

To seed the *Customer and Financial Hub* model:

1. In the Convergence for MDM menu, Select **Help > Getting Started > Create Demo Model...**
2. In the **Getting Started Setup** wizard, Select the **Partial Setup** option and then click **Next**.
3. The second wizard screen allows you to rename the model:
   - If you are running this tutorial for the first time, leave the fields unchanged.
   - If you have already seeded a demonstration model in this repository and want to seed the model a second time with a different name, edit the Demo Model Name (for example, enter *CustomerAndFinancialMDM2*). Note that you will have to use this new name for the rest of the tutorial every time you are requested to enter *CustomerAndFinancialMDM* as a value.
4. Click **Finish**.
5. Click **OK** in the confirmation window.

The demonstration model is seeded, and the **Model Edition** view opens.



Congratulations! You have successfully installed and configured Semarchy Convergence for MDM. You can now proceed and work with the *Customer and Financial Hub* model.

Designing the MDM Hub

In this chapter, you will design a master data hub containing employees, customers, contacts and cost centers information. You will also configure the integration rules to augment, validate and consolidate data coming from several distinct source systems and produce certified golden records.

What is a Model?

A Model contains the description of the master data.

A Model in Convergence for MDM is not simply a *physical* data model but a *logical* model. It includes the logical Entity-Relation definition of the objects stored in the hub — the *entities* with their *attributes*. It also includes the various types, constraints and rules that apply to these entities.
The model also contains the definition of the *integration jobs* that run into the platform to create golden data from raw information pushed by *publishers* (third party applications).

In the first part of the tutorial, we will explore the *Customer and Financial Hub* model.

Creating Customized Data Types

Convergence for MDM includes built-in data types for the attributes of the model's entities, such as String, Number, etc.
You can declare customized types that will be reused across the model, including:

- *User-Defined Types* are a restriction of a built-in data type.
- *Complex Types* are composite types made of attributes.
- *List of Values* or *LOV* are list of Code/Label pairs.

User Defined Types

1. Expand the **User-defined Types** node in the Model Edition.
2. Double-click the **GenericNameType** node. The editor for this user-defined type opens.
   - In the **Details** group, you can see that this user-defined type is based on the *String* built-in type, but with a fixed Length of 80 characters.
   - Click the **Used in** group (in the left banner of the editor) to see the attributes in the model using this user-defined type. For example, the Employee's First Name and Last Name use this type. If the User-Defined type is modified (for example, if we extend the length to 120 characters), every attribute in the model using this type will automatically benefit from this change.
3. Close this **GenericNameType** editor by clicking the Close (a cross) icon on the editor tab.

Complex Types

A *Complex Type* is a user-created type that contains several attributes.

In this example, we will create a complex type called *SimpleAddressType* to represent a postal address. This type will have the following definition attributes: Address, PostalCode, City and Country. We will reuse the *GenericNameType* user-defined type for some of these definition attributes.

Tip: **Auto fill** is a feature in Convergence for MDM that automatically generates field values such as Labels for objects when you provide their name in *CamelCase*. CamelCase consists in having all words joined without spaces, with each word's initial letter capitalized (Examples: *FirstName*, *ZipCode*, *SubjectArea*). CamelCase is a recommended naming convention for objects in Convergence for MDM. For this tutorial, we will use this naming convention to benefit from the Auto fill feature.

To create the *SimpleAddressType* complex type:

1. Right-click the **Complex Types** node and select **Add Complex Type**
2. In the **Create New Complex Type** wizard, enter the following values:
   - **Name**: *SimpleAddressType*. Note that as the **Auto Fill** box is checked, the **Label** is automatically filled in.
3. Click **Finish** to close the wizard. The **Complex Type: SimpleAddressType** editor opens.
4. In the **Description** field, enter the following description: *Type that can be used to represent a simple address.*
5. Select the **Definition Attributes** group (in the left banner of the editor).
6. Click the **Add Definition Attribute**

   button in the **Definition Attributes** table to add a new attribute to the complex type.
7. In the **Create New Definition Attribute** dialog, set the following properties.
   - **Name**: *Address*
   - **Type**: *GenericNameType [User-defined Type]*
8. Click **Finish** to close the dialog.
9. Click the **Add Definition Attribute**

   button.
10. In the **Create New Definition Attribute** dialog, set the following properties.
    - **Name**: *PostalCode*
    - **Type**: *String [Built-in Type]*
    - **Length**: *20*
11. Click **Finish** to close the dialog.
12. Click the **Add Definition Attribute**

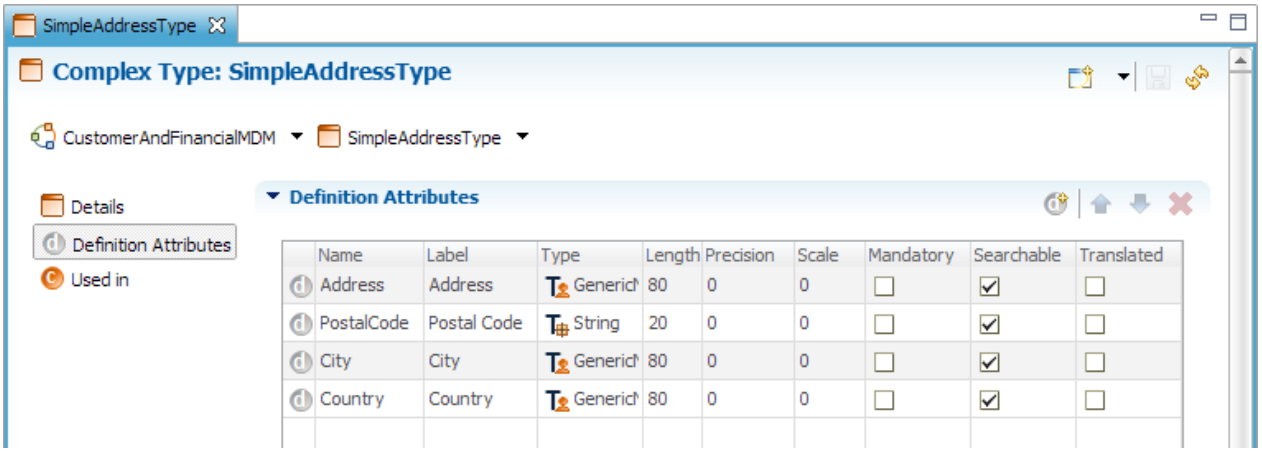
    button.
13. In the **Create New Definition Attribute** dialog, set the following properties.
    - **Name**: *City*
    - **Type**: *GenericNameType [User-defined Type]*
14. Click **Finish** to close the dialog.
15. Click the **Add Definition Attribute**

    button.
16. In the **Create New Definition Attribute** dialog, set the following properties.
    - **Name**: *Country*

- **Type**: *GenericNameType [User-defined Type]*

17. Click **Finish** to close the dialog.
18. Select **File > Save** in the menu. You can alternately press the **Save** button in the toolbar or **CTRL+S** to save the editor.
19. Close the **Complex Type: SimpleAddressType** editor.



Convergence for MDM also models how artifacts are displayed. The *Display Name* for *SimpleAddressType* will define how an address stored in this type is displayed in a compact format in the user interface.

To create a display name for the *SimpleAddressType* type:

1. Expand the **Complex Types** node in the Model Edition.
2. Right-Click the **SimpleAddressType** node in the tree view and then select **Define Display Name**. The **Create New Display Name** wizard appears.
3. In first wizard screen, select the content of the **Separator** field and replace it with a space.
4. Click **Next**
5. In the **Display Name Attributes** page, click the **Add All >>** button to add all available attributes to the **Selected Attributes**. Use the **Move Up** and **Move Down** buttons to arrange them in the following order: *Address*, *Postal Code*, *City* and *Country*.
6. Click **Finish** to close the wizard.
7. Press **CTRL+S** to save the **Display Name: SimpleAddressType** editor.
8. Close the editor.

With this display name, an address stored in a *SimpleAddressType* will display as all the selected attributes, separated by a space.

Designing Entities

Creating Entities

An entity represents an object in the MDM hub. For example Customers, Contacts or Parties are entities. Usually, MDM entities are clearly defined and the definition is agreed upon at the corporate level.

In this example, we will create a *Customer* **Entity**, and use the customized data types for its attributes. The *Customer* entity represents for our organization the *companies we are doing business with*. Customer data is scattered and duplicated in various applications. This entity will be designed to de-duplicate (fuzzy matching) and consolidate this data into single golden customer records.

To create the Customer entity:

1. In the **Model Edition** view, expand the 
   Diagrams node.
2. Double-click the 
   MainDiagram node. The **MainDiagram** editor opens. This diagram displays the entities of the model in a graphical way and allows the creation or modification of these entities. This model already contains the *Contact*, *Employee* and *CostCenter* entities.
3. In the **Palette** (on the right side of the editor), click to select the 
   Add Entity tool.
4. Click in the diagram. The **Create New Entity** wizard opens.
5. In the **Create New Entity** wizard first screen, make sure that the **Auto Fill** box is checked and then enter the following:
   - **Name**: *Customer*
   - **Plural Label**: *Customers*
   - Select *Fuzzy Matching* for the **Matching Behavior**.
6. Click **Next**
7. In the **Primary Key Attribute** screen, make sure that the **ID Generation** is set to *Sequence* starting with a value of *1*.
8. Click **Finish** to close the wizard. The new **Customer** entity is added to the diagram.

9. Press **CTRL+S** to save the editor.
10. Double-click the title of the **Customer** entity in the diagram. The **Entity: Customer** editor opens.
11. In this editor, enter the following value in the **Description** Field: *Entity that represents all customers we are doing business with. Customers come from various source applications such as the CRM and Marketing applications*.
12. Press **CTRL+S** to save the editor and leave it open.

To add attributes to the Customer entity:

Now that the *Customer* entity is created, we will add **Attributes** (fields) to this entity, such as the *CustomerName*. Each attribute will be of a certain type (built-in, user-defined, complex, etc.). We will also define whether these attributes are mandatory or not.

1. In the **Entity: Customer** editor, select the

   Attributes section (in the left banner of the editor). The list of attributes is displayed. It already contains the primary key field *CustomerID* defined when we created the entity.
2. In the **Attributes** table, select the

   Add Simple Attribute button.
3. In the **Create New Simple Attribute** dialog, make sure that the **Auto Fill** box is checked and then enter the following:
   - **Name**: *CustomerName*
   - **Type**: *GenericNameType [User-defined Type]*
   - Check the **Mandatory** box.
   - **Mandatory Validation Scope**: *Pre and Post Consolidation*
4. Click **Finish**. The *CustomerName* attribute is created. It is a mandatory attribute for this entity.
5. In the **Attributes** table, select the

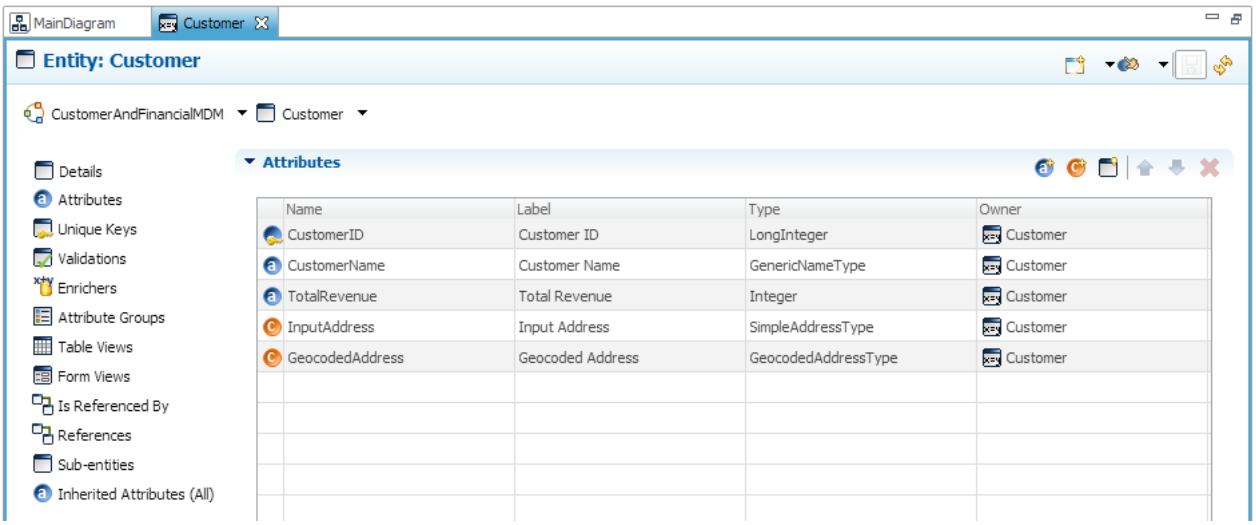
   Add Simple Attribute button.
6. In the **Create New Simple Attribute** dialog, make sure that the **Auto Fill** box is checked and then enter the following:
   - **Name**: *TotalRevenue*
   - **Type**: *Integer [Built-in Type]*
   - Keep the **Mandatory** box unchecked.
7. Click **Finish**.
8. In the **Attributes** table, select the

   Add Complex Attribute button.
9. In the **Create New Complex Attribute** dialog, make sure that the **Auto Fill** box is checked and then enter the following:
   - **Name**: *InputAddress*
   - **Physical Prefix**: *INP*
   - **Complex Type**: *SimpleAddressType [Complex Type]*
10. Click **Finish**. The complex attribute *InputAddress* is now created.
11. In the **Attributes** table, select the

    Add Complex Attribute button.
12. In the **Create New Complex Attribute** dialog, make sure that the **Auto Fill** box is checked and then enter the following:
    - **Name**: *GeocodedAddress*
    - **Physical Prefix**: *GEO*
    - **Complex Type**: *GeocodedAddressType [Complex Type]*
13. Click **Finish**.
14. Press **CTRL+S** to save the **Entity: Customer** editor and leave it open.



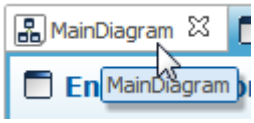
Note: The **Physical Table Name**, **Physical Column Name** and **Physical Prefix**, define the name of the physical database objects created for the entities and attributes. As this tutorial includes loading scripts targeting predefined database objects, it is important to use the correct physical prefix and names when designing the model, in order to deploy physical database objects with the expected names.

To create a display name for the Customer entity:

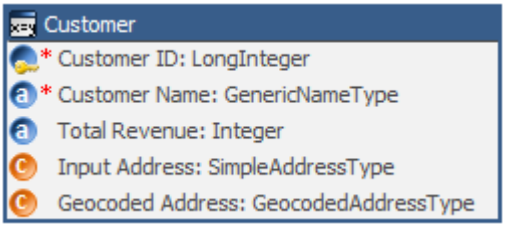
We will now define the **Display name** for the entity, which is the way to display it in a compact way (for example, in a tree view). For customers, we only want to show the *CustomerName*.

1. Select the **Customer** node in the **Outline** view (right of the screen), right-click and then select **Define Display Name**. The **Create New Display Name** wizard appears.
2. In first wizard screen, select the content of the **Separator** field, and replace it with a space.
3. Click **Next**
4. In the **Display Name Attributes** screen, select the *CustomerName* attribute and click the **Add >>** button to add it to the **Selected Attributes**.
5. Click **Finish** to close the wizard.
6. Press **CTRL+S** to save the **Entity: Customer** editor and leave it open.

You can now return to the **MainDiagram** editor by selecting it.



The diagram now displays the Customer entity with its various attributes as shown below.



Congratulations! You have successfully created your first entity!

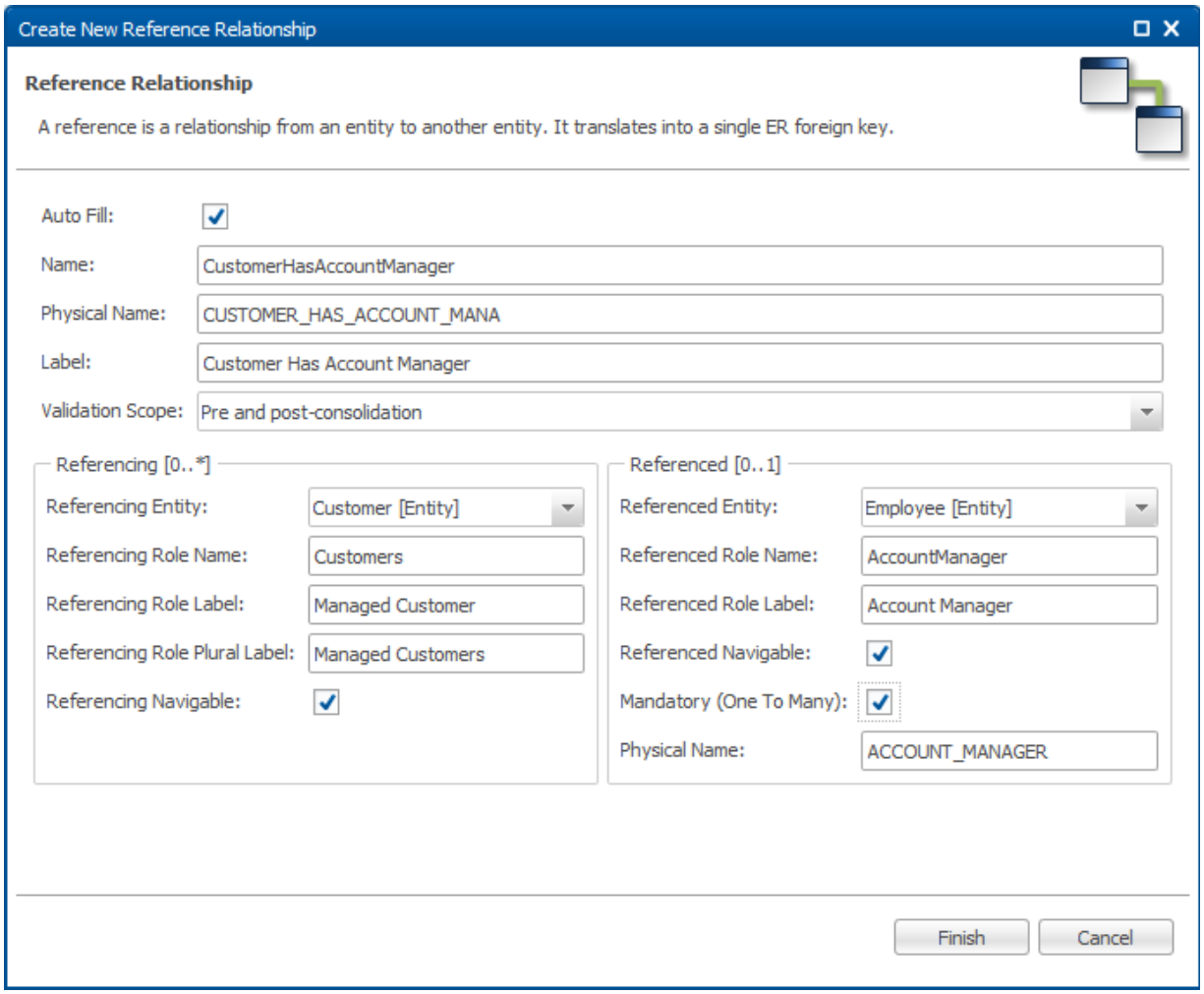
Tip: Attributes can also be created directly in the diagram by right-clicking on any of the entities and selecting the appropriate actions.

Creating References

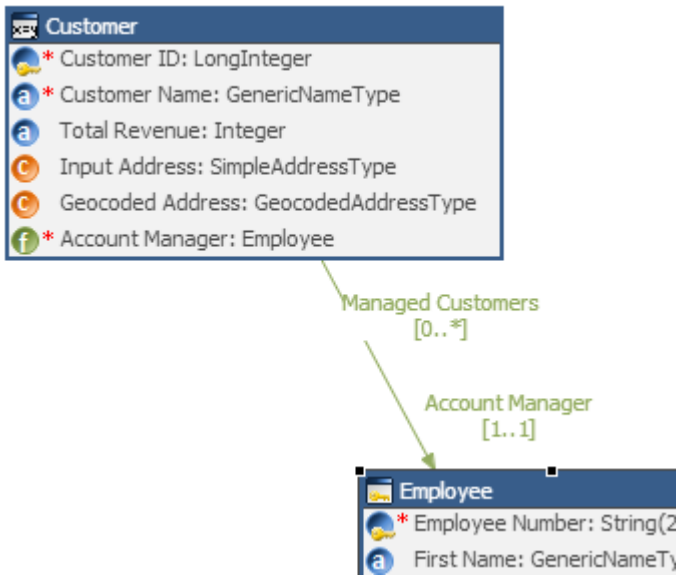
Entities are related using **References Relationships**. We will now create the following relations on the newly created *Customer* entity:
- *CustomerHasAccountManager*, representing the fact that a *Customer* is managed by one *Employee* (his Account Manager).
- *ContactBelongsToCustomer*, representing the fact that a *Contact* (a person) is directly attached to a *Customer* (a company).

To create the *CustomerHasAccountManager* relation:

1. In the **Palette** of the **MainDiagram** editor, click to select the  **Add Reference** tool.
2. Click and drag from **Customer** to **Employee**. The **Create New Reference Relationships** wizard opens.
3. In the **Create New Reference Relationships** wizard screen, make sure that the **Auto Fill** box is checked and then enter the following:
   - **Name**: *CustomerHasAccountManager*
   - **Validation Scope**: *Pre and Post Consolidation*
   - In the **Referencing** group, check that **Referencing Entity** is set to *Customer [Entity]*.
   - In the **Referencing** group, set the following values:
     - **Referencing Role Name**: *Customers*
     - **Referencing Role Label**: *Managed Customer*
     - **Referencing Role Plural Label**: *Managed Customers*
   - In the **Referenced** group, check that **Referenced Entity** is set to *Employee [Entity]*.
   - In the **Referenced** group, set the following values:
     - **Referenced Role Name**: *AccountManager*
     - Select the **Mandatory (One to Many)** option.
4. Review the content of the wizard.



5. Click **Finish** to close the wizard. The reference appears now in the diagram as an arrow linking the *Customer* entity to the *Employee* entity.

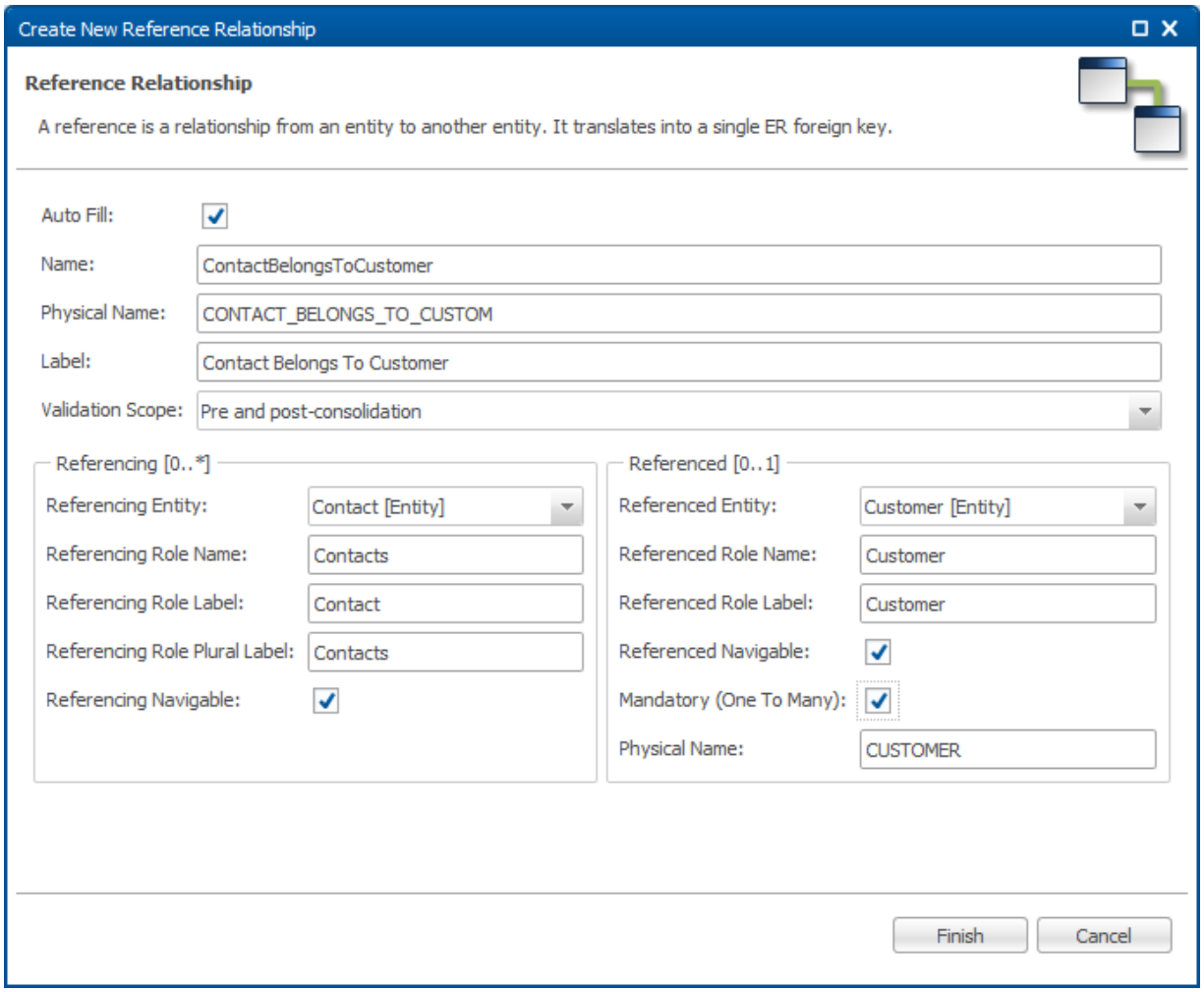


6. Press **CTRL+S** to save the editor.

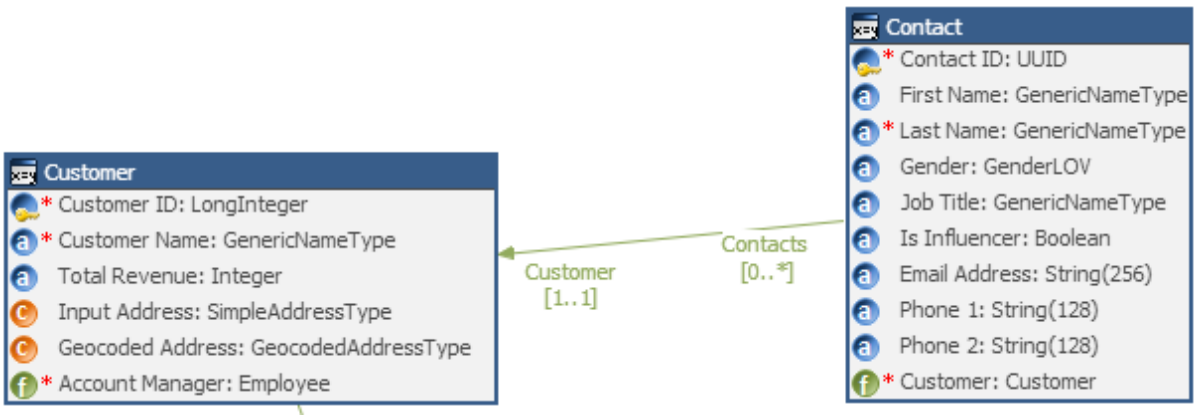
This reference has a *Pre and Post Consolidation* **Validation Scope**: On the source data, but also after consolidating the candidate customer golden records, we will check that customers reference valid account managers. As the reference is marked as **Mandatory**, we will not allow null values. Customer records that do not respect these rules will be rejected as errors.

To create the *ContactBelongsToCustomer* relation:

1. In the **Palette** of the **MainDiagram** editor, click to select the
   
   Add Reference tool.
2. Click and drag from **Contact** to **Customer**. The **Create New Reference Relationships** wizard opens.
3. In the **Create New Reference Relationships** wizard screen, make sure that the **Auto Fill** box is checked and then enter the following:
   - **Name**: *ContactBelongsToCustomer*
   - **Validation Scope**: *Pre and post-consolidation*
   - In the **Referencing** group, check that **Referencing Entity** is set to *Contact [Entity]*.
   - In the **Referencing** group, set the following values:
     - **Referencing Role Name**: *Contacts*
     - **Referencing Role Label**: *Contact*
     - **Referencing Role Plural Label**: *Contacts*
   - In the **Referenced** group, check that **Referenced Entity** is set to *Customer [Entity]*.
   - In the **Referenced** group, select the **Mandatory (One to Many)** option.
4. Review the content of the wizard.



5. Click **Finish** to close the wizard. The reference appears now in the diagram as an arrow linking the *Contact* entity to the *Customer* entity.

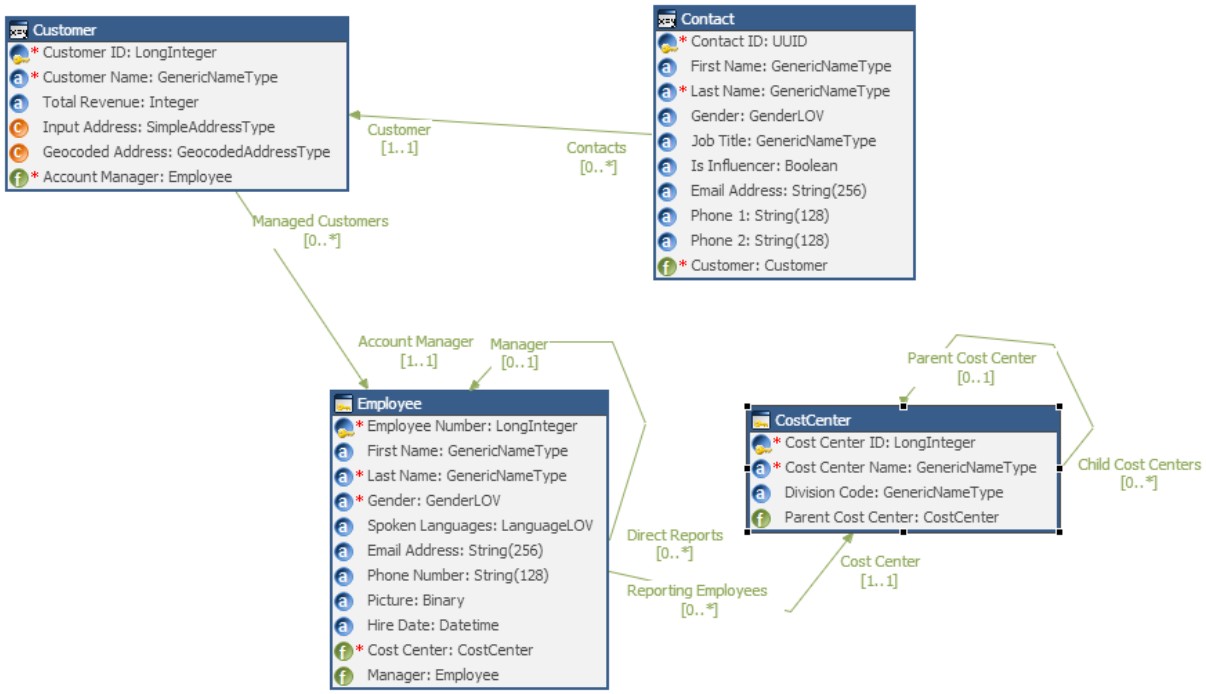


6. Press **CTRL+S** to save the editor.

This reference is marked as **Mandatory** and its **Validation Scope** is *Pre and post-consolidation*: We will check that contacts are always attached to valid customers in the source records as well as in the candidate golden records.

Review the Diagram

You can rearrange the diagram elements as shown below, and close the diagram editor.



Congratulations! You have now related the *Customer* entity to the rest of the logical model.

Defining Constraints

While creating attributes and references, we have introduced data quality rules (mandatory attributes and references that must point to a valid record). We will now create a **Validation** rule to enforce a certain level of completeness on the customer's postal address.

Constraints add validation rules on entities, which can run while the Hub refines information from a variety of sources into golden data.
Constraints may be mandatory attributes, unique keys, validations, list of values, etc. Reference relationships are also constraints as they may define referential integrity between entities.

Constraints are checked at various points of the integration process:

- A **Pre-Consolidation** validation checks the records before they are matched, de-duplicated and consolidated. Only records that meet these validations participate in the matching/de-duplication and consolidation phases.
- A **Post-Consolidation** validation validates the records after they are de-duplicated and consolidated, and before they become golden.

To add a validation:

1. Return to the *Customer* entity editor.
2. In the **Outline** view, expand the **Customer** Node.
3. In the **Outline** view, right-click the **Validations** node and then select **Add SemQL Validation**
4. In the **Create New SemQL Validation** wizard, enter the following:
   - **Name**: *ValidateAddressCompleteness*
   - **Label** is auto filled with: *Validate Address Completeness*
   - **Description**: *Source address should contain at least one address line and either a postal code or a city.*
   - **Condition**: `InputAddress.Address is not null and ( not ( InputAddress.PostalCode is null and InputAddress.City is null ) )`
   - **Validation Scope**: *Pre-Consolidation Only*
5. Click **Finish** to create the validation. You return to the **Entity: Customer** editor.
6. Press **CTRL+S** to save this editor and leave it open.

Note that the validations, as well as the transformation expressions in Convergence for MDM, are defined using the SemQL Language, which is executed within the Oracle Database engine. Convergence for MDM leverages the transformation capabilities and processing power of the database hosting the MDM Hub, providing the best performances for building golden records from very large source data volumes.

Understanding the Certification Process

Source data is published to the MDM Hub from enterprise applications (for example via an ETL tool), or entered by users through human workflows.
This information is frequently incomplete, inconsistent, not standardized, and contains data that violates some of the constraints or restrictions defined for the entities. There is also duplicate data that must be matched, de-duplicated and consolidated into the golden records.

The certification process is a key element of master data management. It uses the various rules defined in the model to refine imperfect source data into golden data.

Enriching Source Data

The process that certifies source data into golden data uses the data quality rules previously defined in the tutorial. It must also include **Enrichers** to augment and standardize the source data.
In this example, we will create several *Enrichers*:
- A **SemQL Enricher** that uses the SemQL language and the Oracle Database to enrich data submitted by the publishers for the Customer entity.
- A **Plug-in Enricher** that enriches and standardizes the employees' phone numbers.

- A **Plug-in Enricher** that uses the Google Maps API for enriching the customer addresses with geographical information.
Both plug-in enrichers illustrate the Convergence for MDM extensibility framework.

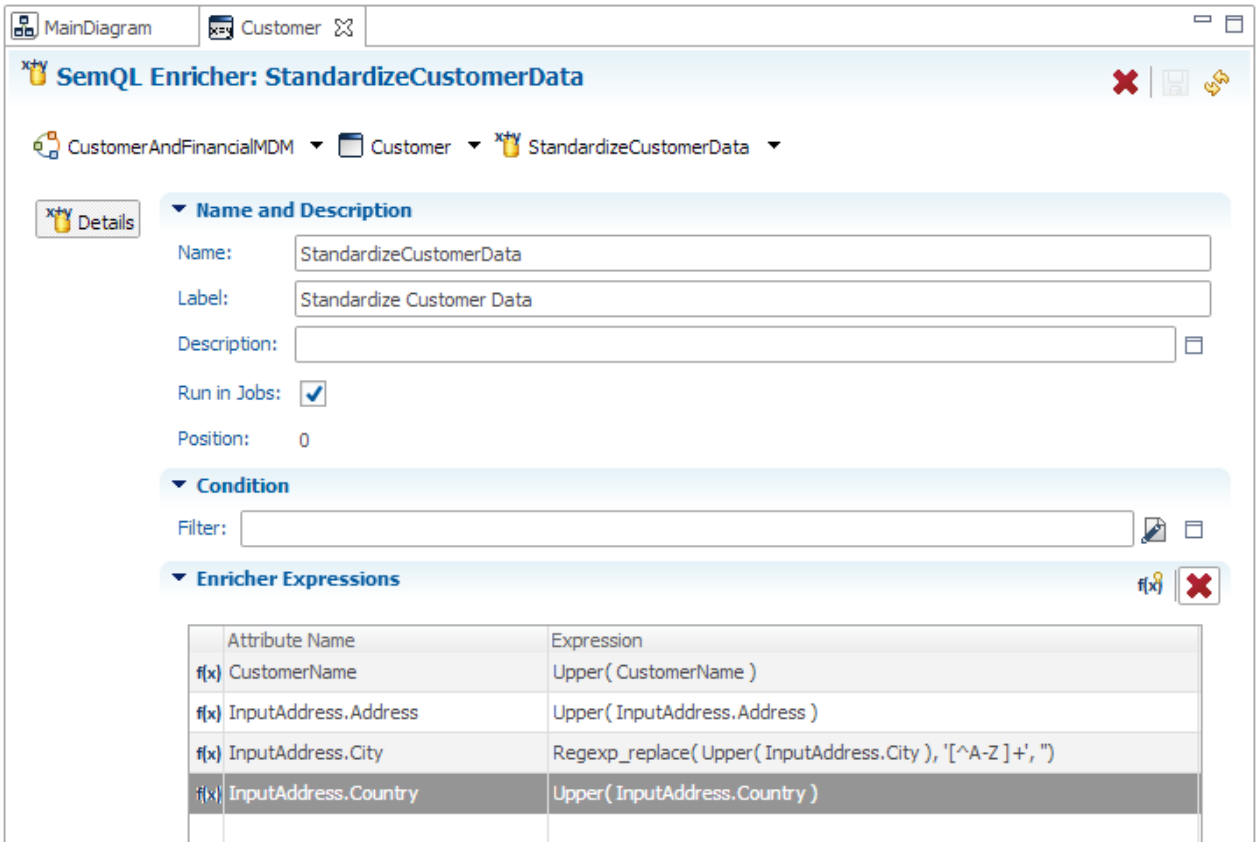
Creating a SemQL Enricher

To create a SemQL enricher:

1. In the **Outline** view, right-click the **Enrichers** node and then select **Add SemQL Enricher**
2. In the **Create New SemQL Enricher** wizard, enter the following:
   - **Name**: *StandardizeCustomerData*
3. Click **Next**
4. In the **Enricher Expressions** screen, double-click the *Customer Name* line in the **Available Attributes** list. It is added to the **Used Attributes** list.
5. Repeat the previous operation to add *InputAddress.Address*, *InputAddress.City* and *InputAddress.Country* to the **Used Attributes** list.
6. Click **Finish**. The **SemQL Enricher: StandardizeCustomerData** editor appears. This editor contains the list of attributes to enrich in the **Enricher Expressions** table, but there is no expression to enrich them.
7. Define the expression for *CustomerName*:
   1. In the **Enricher Expression** table, select the **Expression** column for the *CustomerName* attribute and then click the

      Edit Expression button in the cell. The **SemQL Editor** appears.
   2. In the **Expression** field, enter: `Upper( CustomerName )`
   3. Click **OK** to close the SemQL editor.
8. Repeat these operations to create the following Enricher Expressions:
   - InputAddress.Address: `Upper( InputAddress.Address )`
   - InputAddress.City: `Regexp_replace( Upper( InputAddress.City ), '[^A-Z ]+', '')`
   - InputAddress.Country: `Upper( InputAddress.Country )`
9. Press **CTRL+S** to save this editor and leave it open.



Creating a Plug-in Enricher

Plug-in Enrichers process information provided to them as *Inputs* and deliver *Outputs*. When using such an enricher, you need to map on the plug-in input and output the attributes of the entity to be enriched.

For example, the *Google Maps Enricher* that is used in this example takes as an input a basic address information (address line, city name, country and postal code) and outputs a fully geocoded address (street name/number, longitude/latitude, etc.) .

Enriching Postal Addresses

> **Note on the Postal Address Enricher**
> This enricher uses the Google Maps Service and requires an Internet connection. Creating this enricher is an optional step. If you do not have access to the Internet, you can skip this step and go to the Enriching Phone Numbers step. Per Google's terms of service, there is a limitation to the number of requests you can perform on this service. Please refer to the Google Site for more information.

To create a plug-in enricher for addresses:

1. In the **Outline** view, right-click the **Enrichers** node and then select **Add Plug-in Enricher**

2. In the **Create New Plug-in Enricher** wizard, enter the following:
   - **Name**: *GeocodeInputAddressUsingGoogle*
   - **Label** is auto filled with: *Geocode Input Address Using Google*
   - **Plug-in ID**: *Google Maps Enricher – com.semarchy.integration...*
3. Click **Finish** to create the Plug-in Enricher. The **Plug-in Enricher: GeocodeInputAddressUsingGoogle** editor appears.
4. In the **Description** field, enter: *Illustrates the use of the Google Maps Geocoder Plug-in.*
5. Deselect the **Run in Jobs** box if you are not connected to the Internet.
6. Now, you must provide values for the plugin input parameters. To map the *Address Line* plug-in input:
   1. In the **Plug-in Inputs** table, select the **Expression** column for the *Address Line* input and then click the

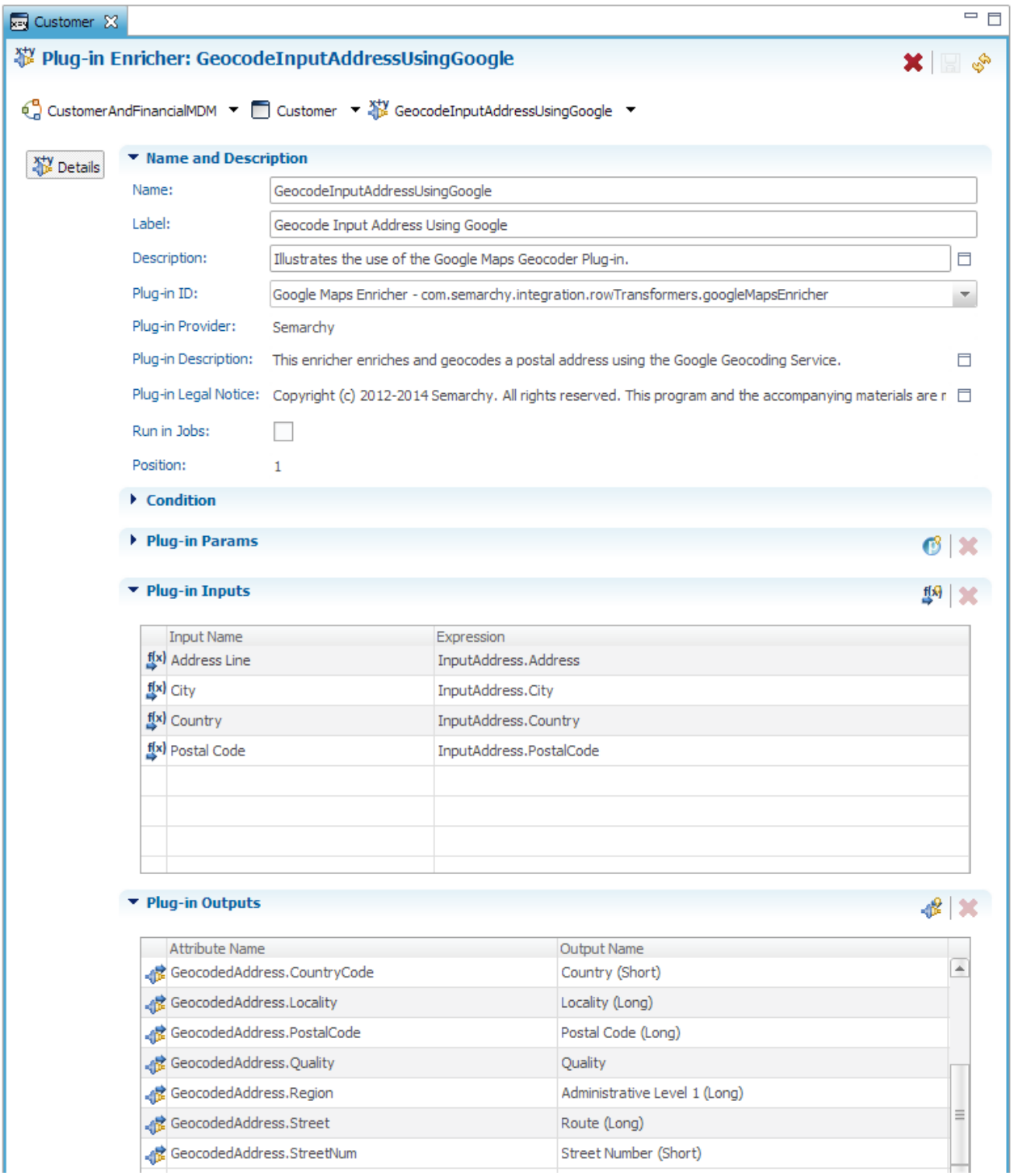
      Edit Expression button in the cell. The **SemQL Editor** appears.
   2. In the **Expression**, enter `InputAddress.Address`. You can alternately double-click this attribute in the **Attributes** list available on the left side of the editor to add it to the expression.
   3. Click **OK** to close the SemQL Editor.
7. Repeat the same operation to map the other Plug-in inputs with the following attributes:
   - *City* : `InputAddress.City`
   - *Country*: `InputAddress.Country`
   - *Postal Code*: `InputAddress.PostalCode`
8. Map the plug-in outputs:
   1. Scroll down in the Editor, and in the **Plug-in Outputs** table, click the

      Define Plug-in Outputs button. The **Define Output Bindings** dialog appears.
   2. In this dialog, select all the attributes starting with *GeocodedAddress* line in the **Available Attributes** list and then click the **Add >>** button. All these attributes are added to the **Used Attributes** list.
   3. Click **Finish** to close the dialog. The attributes from the entity that were selected now appear in the **Plug-in Outputs** table. They must be mapped to the output of the plug-in.
9. Define the bindings for the *GeocodedAddress.Region* attribute.
   1. In the **Plug-in Outputs** table, click the **Output Name** column on the *GeocodedAddress.Region* plug-in output line. This column is a drop-down list from which you can now select *Administrative Level 1 (Long)*.
10. Repeat the same operation to map the other attributes of the *GeocodedAddress* complex type attribute of the *Customer* entity on the plug-in outputs according to the list below:
    - GeocodedAddress.BoxEast: *East bound longitude*
    - GeocodedAddress.BoxNorth: *North bound latitude*
    - GeocodedAddress.BoxSouth: *South bound latitude*
    - GeocodedAddress.BoxWest: *West bound longitude*
    - GeocodedAddress.CoordLatitude: *Latitude*
    - GeocodedAddress.CoordLongitude: *Longitude*
    - GeocodedAddress.Country: *Country (Long)*
    - GeocodedAddress.CountryCode: *Country (Short)*
    - GeocodedAddress.Locality: *Locality (Long)*
    - GeocodedAddress.PostalCode: *Postal Code(Long)*

- GeocodedAddress.Quality: *Quality*
- GeocodedAddress.Street: *Route (Long)*
- GeocodedAddress.StreetNum: *Street Number (Short)*

11. Press **CTRL+S** to save this editor and leave it open.


Customer
Plug-in Enricher: GeocodeInputAddressUsingGoogle
CustomerAndFinancialMDM
Customer
GeocodeInputAddressUsingGoogle
Details
Name and Description
Name: GeocodeInputAddressUsingGoogle
Label: Geocode Input Address Using Google
Description: Illustrates the use of the Google Maps Geocoder Plug-in.
Plug-in ID: Google Maps Enricher - com.semarchy.integration.rowTransformers.googleMapsEnricher
Plug-in Provider: Semarchy
Plug-in Description: This enricher enriches and geocodes a postal address using the Google Geocoding Service.
Plug-in Legal Notice: Copyright (c) 2012-2014 Semarchy. All rights reserved. This program and the accompanying materials are r
Run in Jobs:
Position: 1
Condition
Plug-in Params
Plug-in Inputs
Input Name
Expression
Address Line
InputAddress.Address
City
InputAddress.City
Country
InputAddress.Country
Postal Code
InputAddress.PostalCode
Plug-in Outputs
Attribute Name
Output Name
GeocodedAddress.CountryCode
Country (Short)
GeocodedAddress.Locality
Locality (Long)
GeocodedAddress.PostalCode
Postal Code (Long)
GeocodedAddress.Quality
Quality
GeocodedAddress.Region
Administrative Level 1 (Long)
GeocodedAddress.Street
Route (Long)
GeocodedAddress.StreetNum
Street Number (Short)

Note on the Phone Number Enricher
This enricher uses a built-in library that parses and standardizes international and national phone numbers and optionally infer geographical information from the phone numbers. For this example, we will standardize the phone numbers from the US employees in the national format '(ddd) ddd-dddd'.

To create a plug-in enricher for employee phone numbers:

1. In the **Model Edition** view, expand the **Entities > Employee** node.
2. Select the **Enrichers** node, right-click and then select **Add Plug-in Enricher**.
3. In the **Create New Plug-in Enricher** wizard, enter the following:
   - **Name**: *StandardizeEmployeePhoneNumber*
   - **Label** is auto filled with: *Standardize Employee Phone Number*
   - **Plug-in ID**: *Convergence Phone Enricher – com.semarchy.engine...*
4. Click **Finish** to create the Plug-in Enricher. The **Plug-in Enricher: StandardizeEmployeePhoneNumber** editor appears.
5. In the **Description** field, enter *Standardizes the Employee phone number to the US national format using the built-in plugin.*
6. Now, you must provide values for the plugin input parameters. To map the *Input Phone Number* plug-in input:
   1. In the **Plug-in Inputs** table, select the **Expression** column for the *Input Phone Number* input and then click the

      Edit Expression button in the cell. The **SemQL Editor** appears.
   2. In the **Expression**, enter `PhoneNumber`. You can alternately double-click this attribute in the **Attributes** list available on the left side of the editor to add it to the expression.
   3. Click **OK** to close the SemQL Editor.
7. Repeat the previous step to perform the following mappings:
   - *Region Code*: `'US'`. This input indicates to the enricher that the input phone numbers are from the US.
   - *Enriched Phone Format*: `'NATIONAL'`. This input indicates to the enricher that the output phone numbers should be formatted in national format: '(ddd) ddd-dddd'.
   - Select the *Region of Origin* plug-in input and then click the

     Delete button to remove it as it is unused.
8. Map the plug-in outputs:
   1. Scroll down in the Editor, and in the **Plug-in Outputs** table, click the

      Define Plug-in Outputs button. The **Define Output Bindings** dialog appears.
   2. In this dialog, select the *PhoneNumber* attribute in the **Available Attributes** list and then click the **Add >>** button. This attribute is added to the **Used Attributes** list.
   3. Click **Finish** to close the dialog. The *PhoneNumber* attribute now appears in the Plug-in Outputs table. It must be mapped to the output of the plug-in.

4. In the **Plug-in Outputs** table, click the **Output Name** column on the *PhoneNumber* plug-in output line. This column is a drop-down list from which you can now select *Enriched Phone Number*.

9. Press **CTRL+S** to save this editor and then close it.

De-duplicating Data

Now that we have defined the rules to augment/standardize the source data as well as the rules to check their quality, we will define how to handle duplicate customer records. This operation takes place in two phases: First, the duplicate records are detected in a phase called **Matching**. Afterwards, information from all these duplicates is assembled in a single master record. This second phase is called **Consolidation**.

Matching Records

Record matching consists of identifying two or more records that are duplicates. For example, an employee with a typo in his name in one of the applications may cause a duplicate in the golden data. Detecting this employee as a duplicate is key to obtaining real unique records in the MDM Hub. This matching process requires a preliminary *binning* mechanism to accelerate the matching process. Binning consists of creating smaller sub-sets of records. Matching will take place on these subsets to avoid matching every record against the whole data set.

To create a matcher:

1. In the **Outline** view, right-click the **Customer > Matcher** node and then select **Define SemQL Matcher**
2. In the **Create New SemQL Matcher** wizard, enter the following:
   - **Description**: *Find customer duplicates using name and address*
3. Click **Finish**. The **SemQL Matcher – Customer** editor appears.
4. In the **Binning Expressions** group, click the

   Add Binning Expression button. The SemQL Editor opens.
5. In the SemQL Editor, enter the following expression: `InputAddress.Country`
6. Click **OK** to close the SemQL Editor. We have now divided the sub sets of possible matches by considering only customers belonging to the same country as possible matching candidates.
7. In the **Matching Condition** group, click the

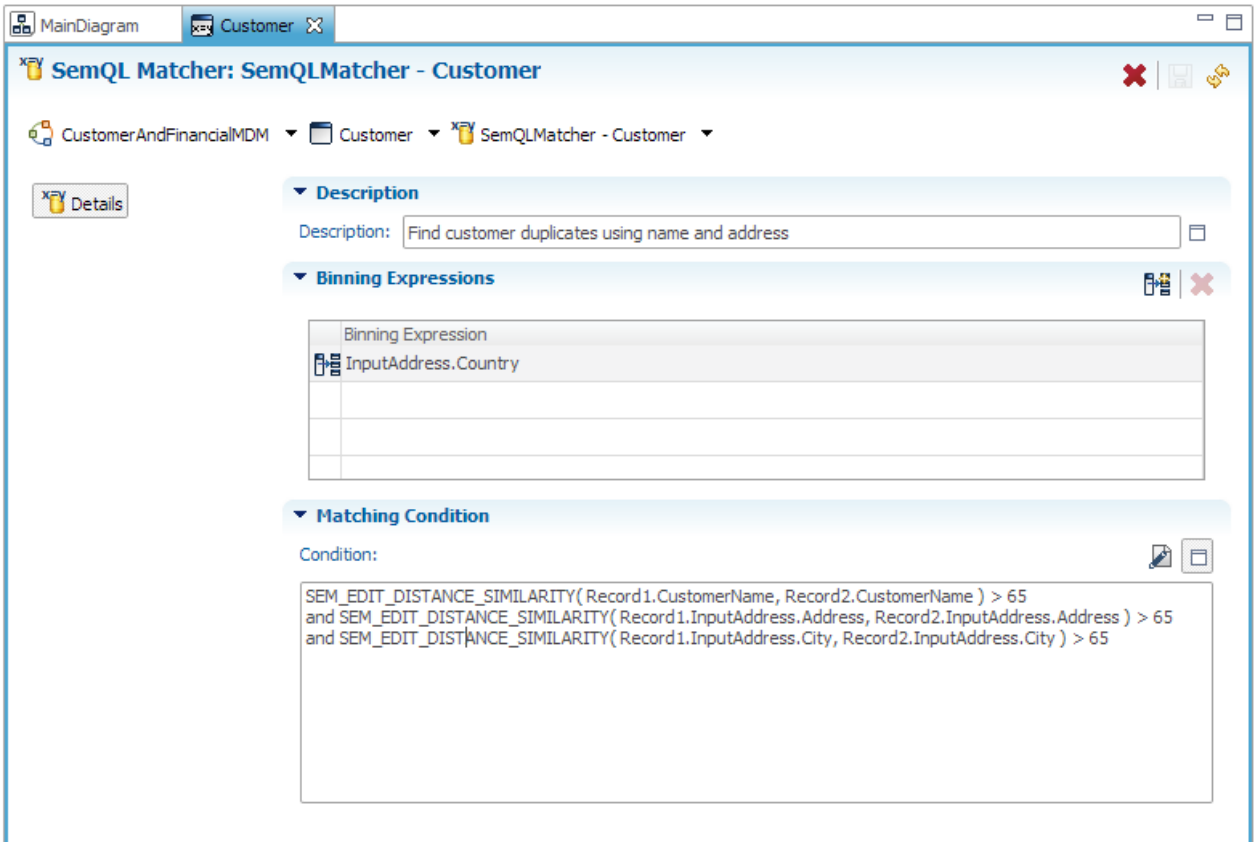
   

   Edit Expression button. The SemQL Editor opens.
8. In the SemQL Editor, enter the following condition:

```
SEM_EDIT_DISTANCE_SIMILARITY( Record1.CustomerName, Record2.CustomerName ) >
65
and SEM_EDIT_DISTANCE_SIMILARITY( Record1.InputAddress.Address,
Record2.InputAddress.Address ) > 65
and SEM_EDIT_DISTANCE_SIMILARITY( Record1.InputAddress.City,
Record2.InputAddress.City ) > 65
```

1. Click **OK** to close the SemQL Editor.

2. Press **CTRL+S** to save this editor and leave it open.



You have created a matching rule that expresses that two customer records would be considered similar if they match all these rules:

1. They belong to the same country (as expressed in the binning expression)
2. They have a strong similarity on their name (greater than 65%) using the Levenshtein distance algorithm:
   `SEM_EDIT_DISTANCE_SIMILARITY( Record1.CustomerName, Record2.CustomerName ) > 65`
3. They have a strong similarity on their address line and city (greater than 65%) using the Levenshtein distance algorithm:
   ```
   SEM_EDIT_DISTANCE_SIMILARITY( Record1.InputAddress.Address,
   Record2.InputAddress.Address ) > 65 and
   SEM_EDIT_DISTANCE_SIMILARITY( Record1.InputAddress.City,
   Record2.InputAddress.City ) > 65
   ```

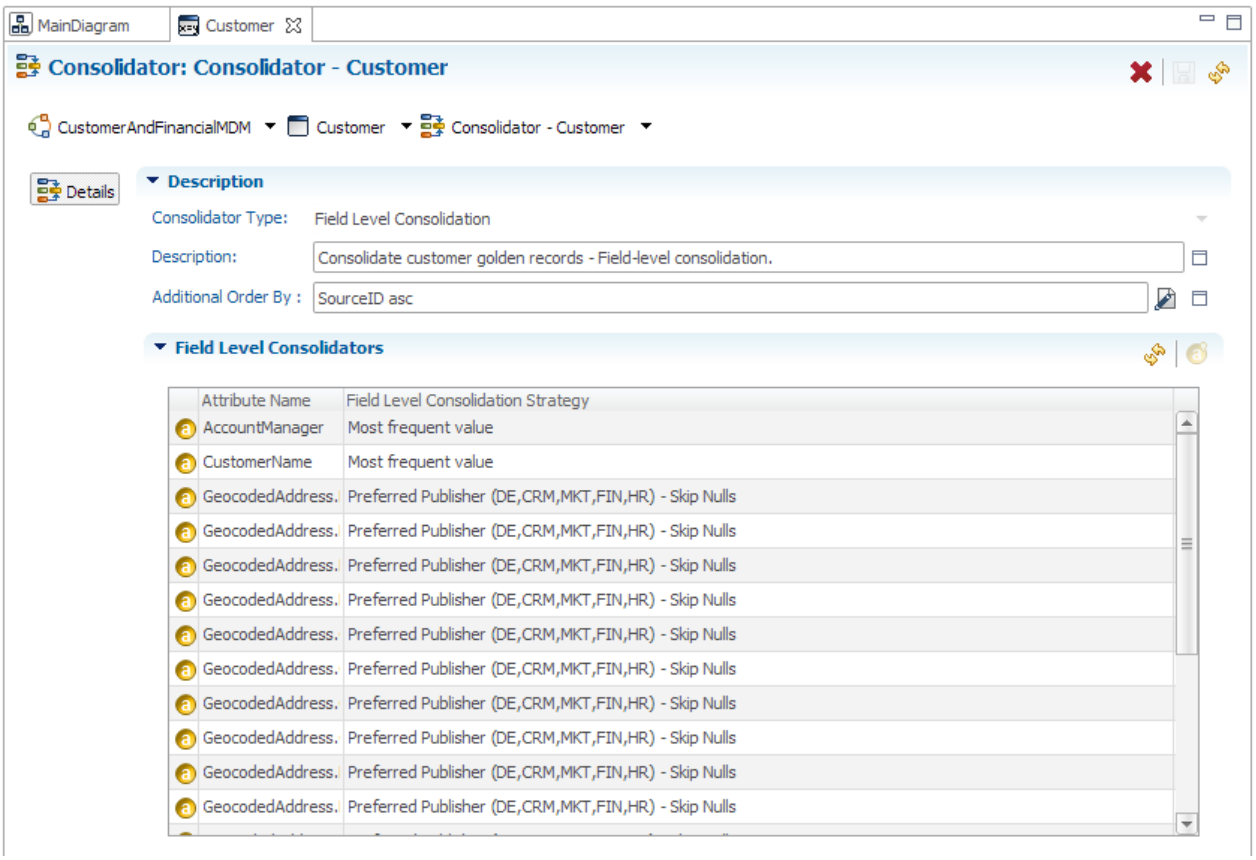
You can of course use other complex rules and matching techniques such as Jaro-Winkler distance, Name normalization or Soundex.

Consolidating Records

When a set of records is identified as a duplicates group, a single **golden** record is created out of it. The *consolidation* process consists of taking the best data from each record to create this golden record. For example, to build a golden customer record, we may take the address and phone number from the shipping application, and the financial numbers from the billing system. The resulting golden record would be a composite of information from several sources.

To create the *Customer* consolidator:

1. Expand **Entities > Customer** in the Model Edition.
2. Right-click the **Consolidator** node and then select **Define Consolidator**
3. In the **Create New Consolidator** wizard, enter the following:
   - **Consolidator Type**: *Field Level Consolidation*. With this type, the consolidation strategy can be different for each attribute.
   - **Description**: *Consolidate customer golden records – Field-level consolidation.*
4. Click **Finish** to close the wizard. The **Consolidator – Customer** editor appears.
5. Define the consolidation strategy for *CustomerName* and *AccountManager*.
   1. In the **Field Level Consolidators** table, press and hold down the control key (CTRL) and then click the *CustomerName* and *AccountManager* attributes to select them.
   2. Click the  **Define Field Level Consolidation Strategy** button.
   3. In the **Define Selected Field-Level Consolidator** wizard, select **Most Frequent Value** and then click **Finish**.
6. Define the consolidation strategy for the geographical attributes.
   1. Select all the other attributes except *TotalRevenue*, that is all attributes starting with *GeocodedAddress* and *InputAddress*.
   2. Click the  **Define Field Level Consolidation Strategy** button.
   3. In the **Define Selected Field-Level Consolidator** wizard, select **Preferred Publisher** and then click **Next**.
   4. In the **Publisher Ranking** table, click the  **Add Publisher Ranking** button.
   5. Double-click the *DataEntry* in the **Available Publishers** list to add to the **Publishers** list.
   6. Repeat this operation to add *CRM*, *Marketing*, *Finance* and then *HumanResources* in this order.
   7. Click **Finish** and then **Finish** again to return to the **Consolidator – Customer** editor.
7. Define the consolidation strategy for *TotalRevenue*.
   1. Double-click the *TotalRevenue* line. The **Define Selected Field-Level Consolidator** wizard appears.
   2. Select **Largest Value** and then click **Finish**.
8. In the **Additional Order By** field, enter `SourceID asc`
9. Press **CTRL+S** to save this editor and leave it open.



You have created a consolidation rule that arbitrates how golden data is created out of the source data when duplicates are detected. In this example, *CustomerName* and *AccountManager* will take the most frequent value from the duplicated records; *Address* information will preferably come from the Data Entry application, then – in order of priority – from the CRM, Marketing, Financial and HR applications. Finally, the *TotalRevenue* will be the largest value from all the duplicates.

As a reference was created on the *Contact* entity to the newly created *Customer*, the *Contact* entity now includes a new field called *Customer*.
We must check that this field is consolidated as expected.

Update the *Contact* Consolidator:

1. Expand the **Entities > Contact > Consolidator** Node.
2. Double-click the **Consolidator – Contact** node. The **Consolidator – Contact** editor appears.
3. In the **Field Level Consolidators** group, click the

   Refresh button to refresh the attributes list and then double-click the *Customer* attribute.
4. In the **Define Selected Field-Level Consolidator** wizard, select **Preferred Publisher** and then click **Next**.

5. In the **Publisher Ranking** table, click the **Add Publisher Ranking** button.
6. Double-click the *DataEntry* in the **Available Publishers** list to add to the **Publishers** list.
7. Repeat this operation to add *CRM*, *Marketing*, *Finance* and then *HumanResources* in this order.
8. Click **Finish** and then **Finish** again to return to the **Consolidator – Contact** editor.
9. Press **CTRL+S** to save this editor and then close it.

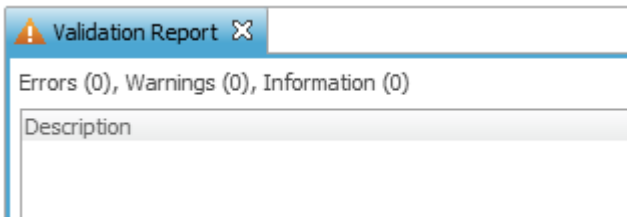
> **Congratulations!** You have defined all the data quality, enrichment and match/merge rules for your *Customer* entity. You have also modified existing entities: A new phone enricher for the *Employees*, and an additional attribute consolidated for the *Contacts*.

Validating the Model

Convergence for MDM detects errors or missing elements in the model. For example, entities with no consolidation strategies will be detected.
Validating a model guides you in the process of designing a model. You can perform regular validations to assess how complete the model really is, and you must validate the model before deploying it.

To validate the model:

1. Close all open editors by right-clicking on any of them and selecting **Close All**.
2. Select the model node **CustomerAndFinancialMDM [0.0]** at the root of the tree.
3. Right-click and select **Validate**.
4. The validation process starts. At the end of the process, the list of issues is displayed in the **Validation Report** view.



> **Note**: At this stage in the tutorial, no error or warning should show up in the validation report.

Working with Integration Jobs

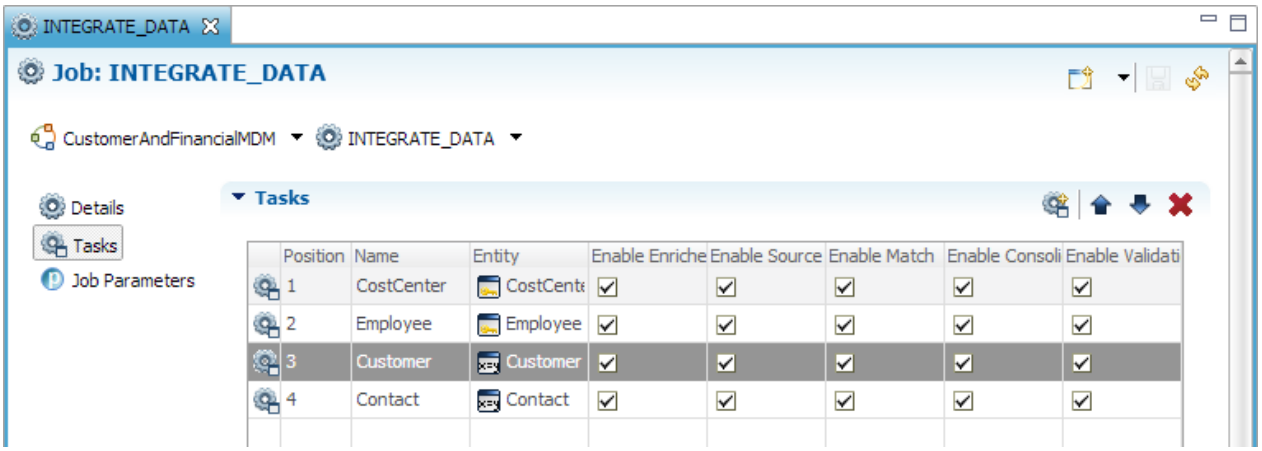
When data is loaded in the hub by an ETL process, or when a user modifies data in the hub as part of a human workflow, Convergence for MDM runs an *Integration Job* to enrich, validate, match/de-duplicate and consolidate this staged data and create golden records. This job uses the enricher, constraints, validations, matchers and consolidators defined in the model.

Integration jobs are already created in the model to certify the master records for the *CostCenter*, *Employee* and *Contacts* entities. As the *Customer* entity was added in the tutorial, it must be added to some of these jobs as a new task.

1. Expand **CustomerAndFinancialMDM [0.0] > Jobs > INTEGRATE_DATA** in the **Model Edition** view.
2. Double-click the **Tasks** node. The **Job: INTEGRATE_DATA** editor opens.

3. Click the

   Add Task button. A **Create New Task** wizard opens.
4. In the **Entity** drop-down box, select *Customer [Entity]*
5. Make sure that all check boxes are checked.
6. Click **Finish** to close the wizard. The *Customer* task is added at the end of the task list.
7. Select this task in the list, and use the **move up** button to move it before the *Contact* task.
8. Press **CTRL+S** to save this editor and then close it.



We also need to modify the job which handles Data Entry/Editing of Customers and Contacts.

1. Expand **CustomerAndFinancialMDM [0.0] > Jobs > DE_CUSTOMER_CONTACT** in the **Model Edition** view.
2. Double-click the **Tasks** node. The **Job: DE_CUSTOMER_CONTACT** editor opens.
3. Click the

   Add Task button. A **Create New Task** wizard opens.
4. In the **Entity** drop-down box, select *Customer [Entity]*
5. Make sure that all check boxes are checked.
6. Click **Finish** to close the wizard. The *Customer* task is added at the end of the task list.
7. Select this task in the list, and use the **move up** button to move it before the *Contact* task.
8. Press **CTRL+S** to save this editor and then close it.

Ordering tasks in jobs is important because validations on certain entities depend on other previously loaded entities. In this example, *Contact* has a reference to *Customer* that mandates loading customer golden records before contacts.

> Congratulations! You have successfully finished your first model! You can now proceed and deploy it to production!

Deploying the MDM Hub

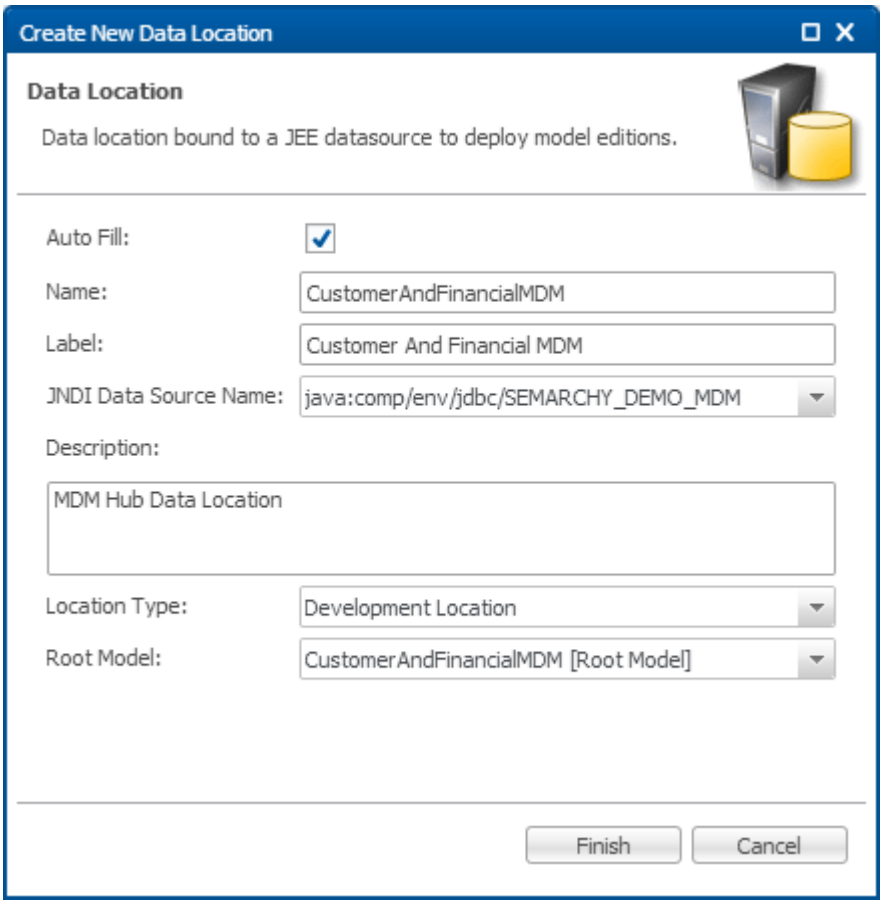
In this chapter, you will deploy your hub to a data location and make it ready for execution.

Creating a Data Location

A *Data Location* is an Oracle schema into which several *Model Editions* and *Data Editions* will be deployed.
A *Model Edition* is a Convergence for MDM model deployed at a given time. An MDM Hub model evolves over time, for example to include new entities or functional areas. Model Editions reflect this evolution in the **structure** of the MDM Hub.
Similarly, a *Data Edition* reflects the evolution of the data stored in the hub over time. You can perform snapshots of the master data at a given time to perform for example a what-if analysis. Data Editions reflect the evolution in the **content** of the MDM Hub.

To create a new data location:

1. In the menu, select **File > New > New Data Location**.
2. In the **Create New Data Location** wizard, enter the following values:
   - **Name**: *CustomerAndFinancialMDM*
   - **JNDI Datasource Name**: *java:comp/env/jdbc/SEMARCHY_DEMO_MDM*
   - **Description**: *MDM Hub Data Location*
   - **Location Type**: *Development Location*
   - **Root Model**: *CustomerAndFinancialMDM [Root Model]*
3. Click **Finish** to close the wizard. The **Data Editions** view opens.

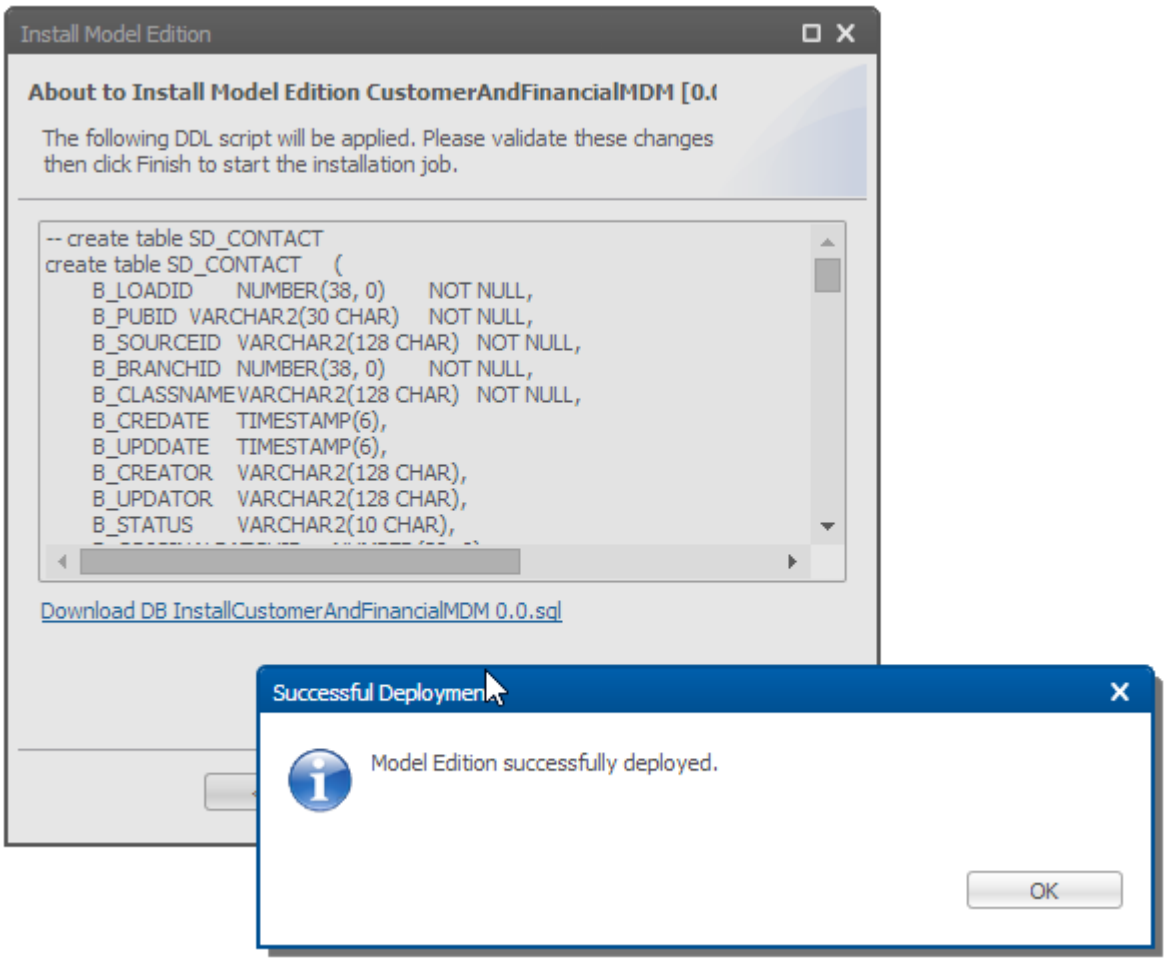


Deploying a Model Edition

Now that the data location is created for the *CustomerAndFinancialMDM* model, it is possible to deploy the first edition of this model.

To install the model edition:

1. In the **Data Editions** view, select the *CustomerAndFinancialMDM* data location, right-click and select **Install Model Edition**. The **Install Model Edition** wizard opens.
2. In **Model Edition**, select *CustomerAndFinancialMDM [0.0] [Model Edition]*
3. Click **Next**. Convergence for MDM generates the SQL code to create the physical database objects corresponding to this model edition. The code is displayed in the wizard and you can download this code to optimize it manually if you wish.
4. Click **Finish** to run this code and deploy the model edition in the data location.
5. Once this operation is complete a **Model Edition Successfully Deployed** dialog appears. Click **OK** to close this dialog.



In the future, when we perform changes to this model's structure, we will be able to create new editions of this model (For example [0.1], [0.2], etc.) and will be able to deploy these new editions in the same data location. These new deployments will generate patching scripts to bring the physical database objects to the level of the new model edition.

Creating a Root Data Edition

To create a data edition:

1. In the **Data Editions** view, select the *CustomerAndFinancialMDM* data location, right-click and select **Create Root Data Edition**. The **Create New Root Data Edition** wizard opens.
2. Click **Next**.
3. In the second wizard screen, for **Deployed Model Edition**, select *CustomerAndFinancialMDM 0.0 [Deployed Model Edition]*.
4. Click **Finish** to create the first data edition.

This data edition is open. It means that we can perform loads and changes to this edition.
Later on, we will be able to close this edition and open a new one. Closing an edition freezes its content, creating a snapshot of the MDM Hub content.

Congratulations! You have successfully deployed your model! You can now proceed and start publishing data!

Running the MDM Hub

In this chapter, you will publish data from various sources in the MDM hub and review the execution of the integration job.

Publish Data to the MDM Hub

Note: For this tutorial, we use a demonstration application that accesses sample data stored in the SEMARCHY_DEMO_SOURCE schema. The tutorial also includes demonstration ETL scripts to load the MDM staging tables from the sample data.

Tip: **Semarchy Convergence for Data Integration** is a high performance data integration platform which can be used in conjunction with Convergence for MDM to publish source data into the MDM hub and consume golden data from the hub. If you want to discover Convergence for DI, a tutorial available on www.semarchy.com/download can drive you through the configuration of Semarchy Convergence for MDM and DI and guide you through designing a data integration project integrating the MDM hub with a variety of information sources, including XML, flat files and databases.

To load sample data into the MDM Hub:

1. In the menu, select **Help > Getting Started > Open Demo Application...**. If you are running the demo application for the first time, click **OK** to install the sample data when prompted. The **Demo Applications** editor opens.
   - The demonstration application contains several editors, each corresponding to an application publishing to the MDM hub. These applications appear as separate tabs at the bottom of the editor.
   - You can edit or delete sample records from these tabs.
   - The **Reset Sample Data** button allows you to restore the sample data to its original state.
   - The **Publish Data to MDM...** button allows you to run the ETL script loading the sample data into the MDM staging table.
2. Click the **Publish Data to MDM...** button in the editor toolbar.
3. In the **Publish Data to MDM** dialog, for **Data Edition** select *CustomerAndFinancialMDM [0.0]*.
4. Click **Finish** to start the loading process.
5. A **Data Submit Success** dialog appears. Click **OK** to close it.

Within 10 seconds, the published data will be detected by the Semarchy Convergence for MDM and integrated by the *INTEGRATE DATA* job.

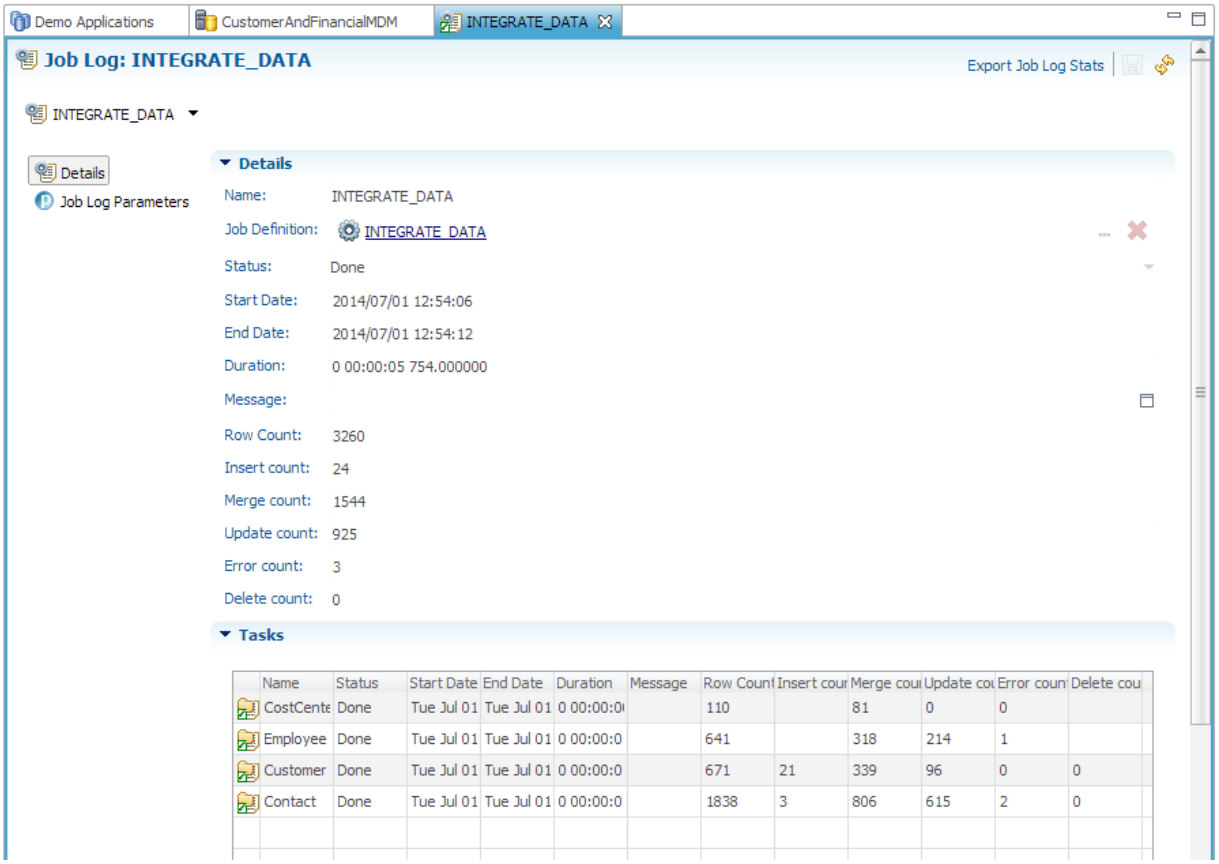
Viewing the Log

The execution of the integration job can be monitored in the log.

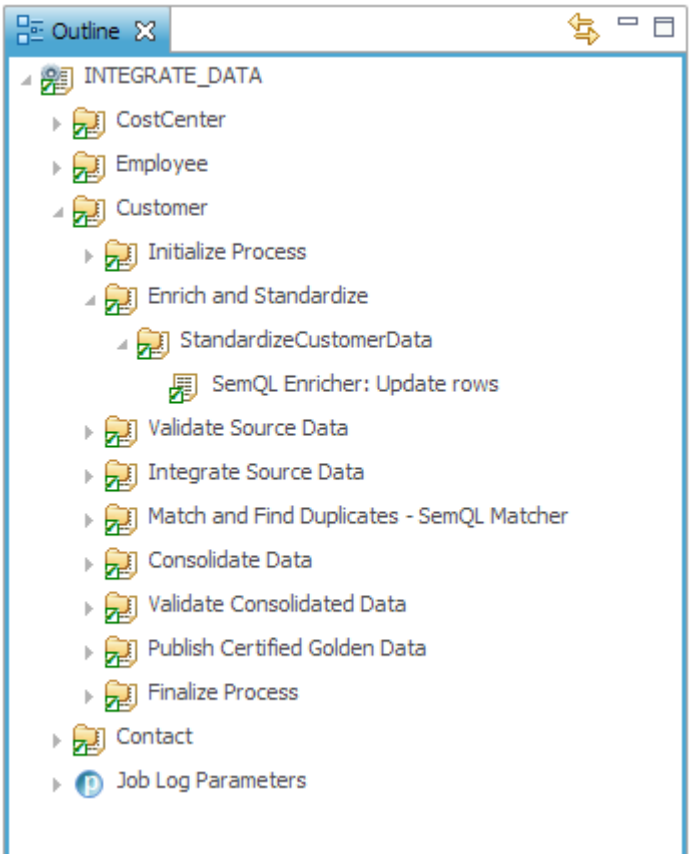
To review the job log:

1. In the **Data Editions** view, expand **CustomerAndFinancialMDM > Deployed Model Editions > CustomerAndFinancialMDM 0.0 > Integration Jobs > INTEGRATE_DATA**.
2. Double-click the **Latest Logs** node under the **INTEGRATE_DATA** job. The list of executed jobs opens.

3. In the **Latest Logs** table, double-click the latest job log (according to the **Start Date**). The **Job Log** editor opens.
4. You can review in this editor the statistics for the job and number of processed rows.
5. Scroll down in the editor to see the list of tasks for this job.



6. In the list of **Tasks**, double-click the *Customer* task. The **Task Group Log: Customer** editor opens.
7. Scroll down to see the **Task Log** table. These tasks represent the successive process the submitted *Customer* has gone through to build golden records.
8. You can also use the **Outline** view to browse a job log. In the **Outline**, expand the **INTEGRATE_DATA > Customer > Enrich and Standardize > StandardizeCustomerData** node. It corresponds to your SemQL enricher.



9. Close all editors by select **File > Close All** in the menu.

Congratulations! You have successfully published and certified your first data batch into the hub! So far, we have modeled the data, the rules, deployed the hub and loaded data into it. It is now time to design a friendly interface for the business users and data stewards.

Creating an Application

Applications empower Business Users and Data Stewards with these abilities:
- Browse and search data
- Author new records
- Edit existing records
- Verify automatically detected duplicates
- Manually match or split records

An application was already created in the model. This application must be modified to support the new *Customer* entity.

Creating Table Views and Form Views

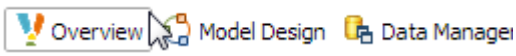
We will start with the creation of the **Views** and **Business Objects**. These determine how data will be displayed to the data stewards and users.

Table Views

First, a new view must be created to define how the new *Customer* entity is displayed in a table layout.

To create a table view for the *Customer* entity:

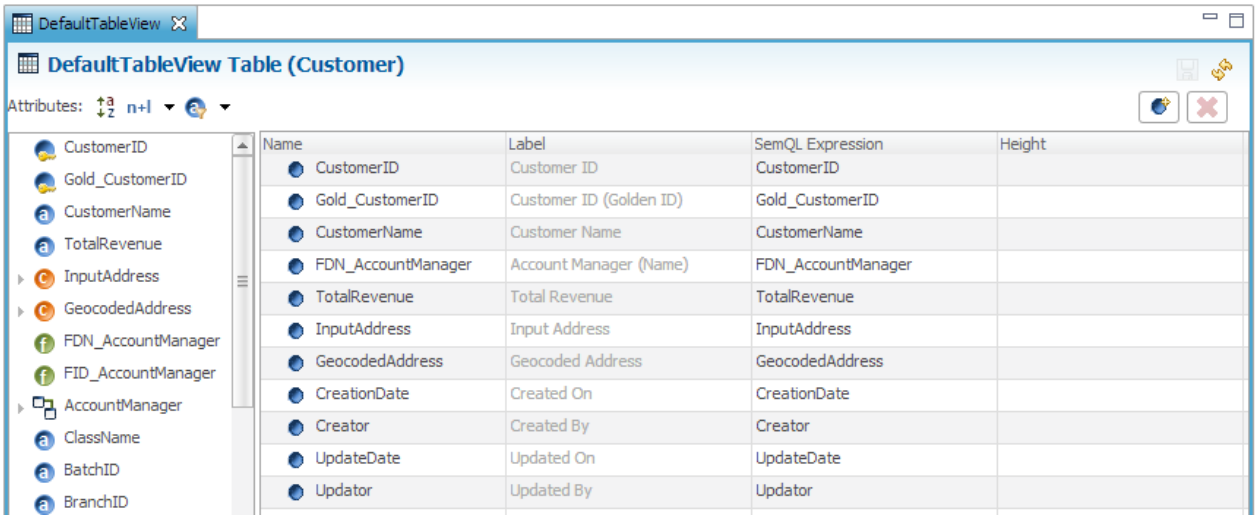
1. In the top-right toolbar, select the **Overview** perspective.



2. In the **Overview** screen, select the **CustomerAndFinancialMDM [0.0]** link. The **Model Design** perspective opens.

3. In the **Model Edition** view, expand the **Entities > Customer** node.
4. Right-click the **Table Views** node and then select **Add Table View**. The **Create New Table View** wizard opens.
5. In the **Create New Table View** wizard, the **Name** is automatically set to *DefaultTableView*.
6. Click **Finish** to close the wizard. The **DefaultTableView Table** editor opens.
7. Press and hold the **CTRL** key to perform a multiple selection and click in the **Attributes** list the following attributes: *CustomerID, Gold_CustomerID, CustomerName, FDN_AccountManager, TotalRevenue, InputAddress, GeocodedAddress, CreationDate, Creator, UpdateDate* and *Updator*.
8. Drag the selected attributes to the table. They are added to the table view, in their order of selection.
9. If needed, reorder the attributes in the table using drag and drop.

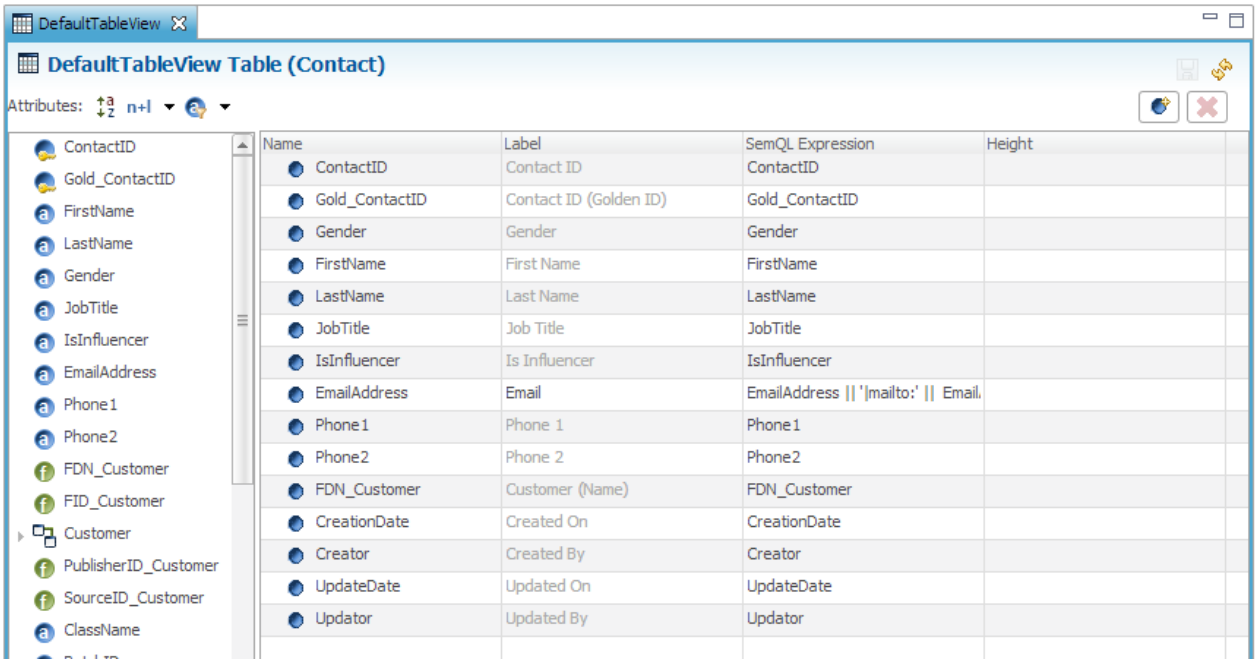
10. Press **CTRL+S** to save the table view and then close the editor.



The *Contact* entity already has a table view into which it is necessary to add the *Customer* this contact belongs to.

To modify the table view for the *Contact* entity:

1. In the **Model Edition** view, expand the **Entities > Contact > Table Views** node.
2. Double-click the **DefaultTableView** node. The **DefaultTableView Table** editor opens.
3. Drag the *FDN_Customer* attribute from the **Attributes** list and drop it in the table between the *Phone2* and *CreationDate* attributes.
4. If needed, reorder the attributes in the table using drag and drop.
5. Press **CTRL+S** to save the editor and then close it.



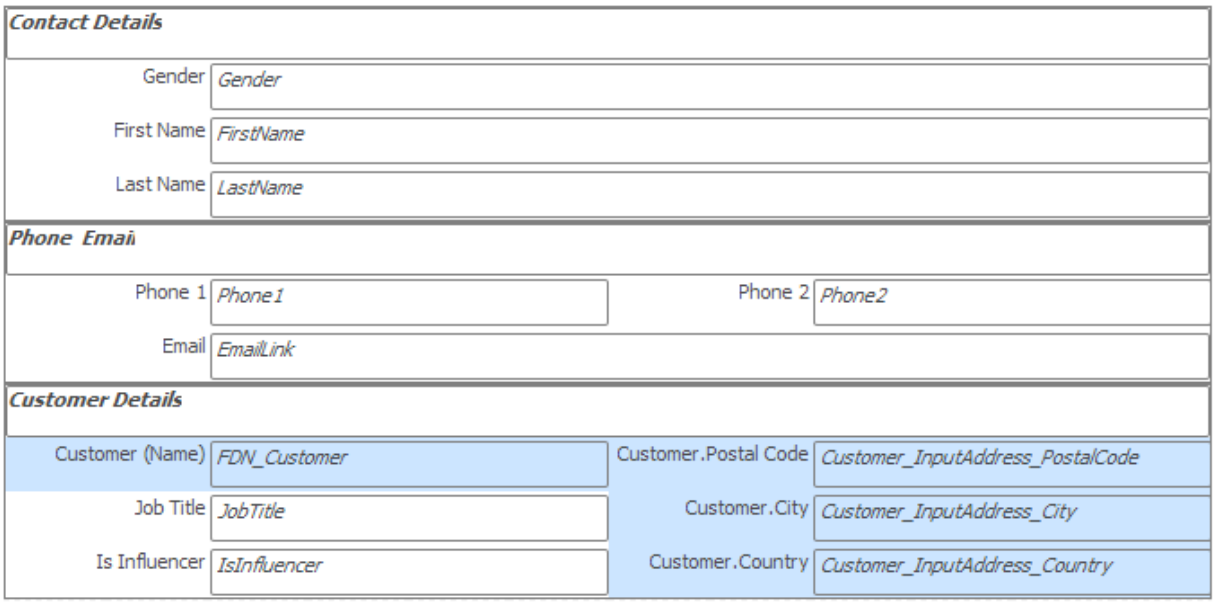
Form Views

Similar changes must be performed on the existing *Contact* form views to extend them with attributes from the new *Customer* entity.

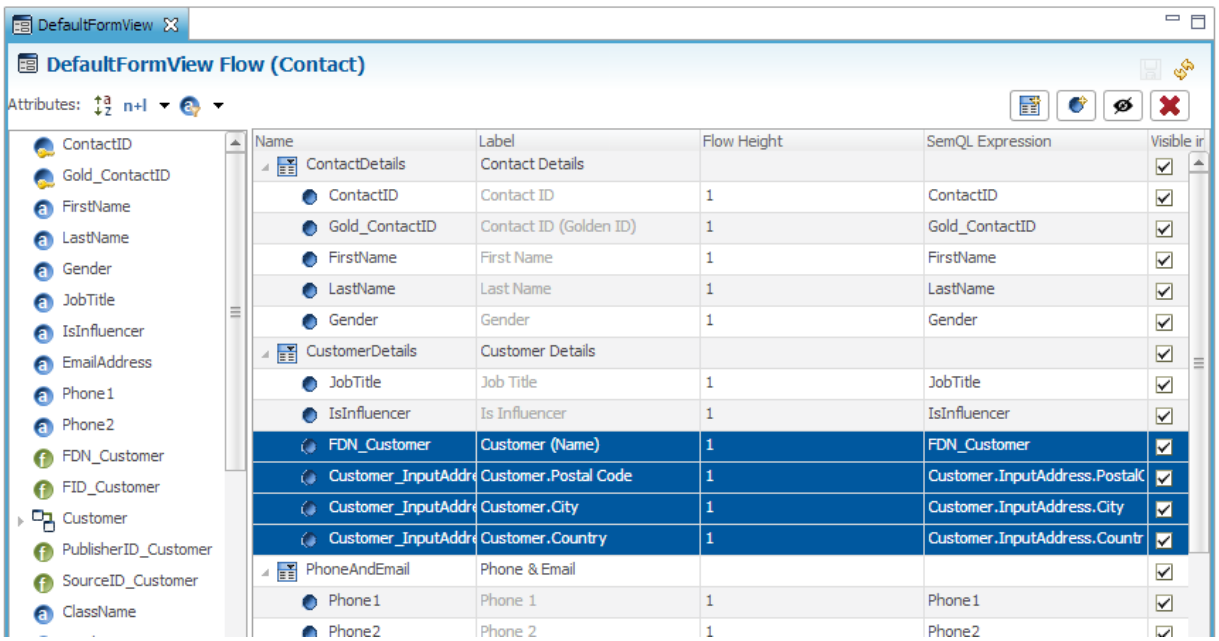
First, the *DefaultFormView* for the *Contact* entity must be extended to include the customer information. This includes the related customer (FDN_Customer), as well as the postal code, city and country of this customer.

1. In the **Model Edition** view, expand the **Entities > Contact > Form Views** node.
2. Double-click the **DefaultFormView** node. The **DefaultFormView** editor opens on the **Grid** tab.
3. Drag and drop the *FDN_Customer* attribute from the **Attributes** list to *Customer Details* section in the grid, above the *Job Title* attribute.
4. Resize the attribute using the
   
   and
   
   buttons and its label using the
   
   and
   
   buttons. You can also move this attribute in the grid by using the buttons from the toolbar or by dragging and dropping the attribute in the grid. The attribute should be aligned with *Job Title*.

5. Drag and drop the *Postal Code*, *City* and *Country* attributes from **Customer > Input Address** in the Attributes list to the *Customer Details* section in the grid. Resize these attributes and their labels to have the form section as shown below.



6. Select the **Flow** tab at the bottom of the editor.
7. The attributes added to the grid should also appear in the flow layout under the *CustomerDetails* section.



8. Press **CTRL+S** to save the form view and then close the editor.

The form view used for data entry in the *Contact* entity must also be modified to include a field to select the customer a given contact is attached to.

1. In the **Model Edition** view, expand the **Entities > Contact > Form Views** node.
2. Double-click the **DataEntryFormView** node. The **DataEntryFormView** editor opens on the **Flow** tab.
3. Drag and drop the *FDN_Customer* attribute from the **Attributes** list to *Customer Details* section in the grid, under the *IsInfluencer* attribute.
4. Press **CTRL+S** to save the form view and then close the editor.

In addition to these changes, form views must be created for the *Customer* entity:

- A *DefaultFormView* used for general purposes (for viewing the Customer in a form)
- A *DataEntryFormView* for entering customer data. The latter has some required fields such as the *SourceID*.

To create the *DefaultFormView* form view:

1. In the **Model Edition** view, select the **Entities > Customer > Form Views**.
2. Right-click and select **Add Form View**. The **Create New Form View** wizard opens.
3. Set the following value in the wizard:
   - **Name**: *DefaultFormView*.
   - **Default Layout**: *Grid Layout*.
   - Select **User Layout Switch** and **Auto Layout Switch**.
4. Click **Finish**. The **DefaultFormView** editor opens on the **Grid** tab.
5. Drag the

   Add Form Section button from the toolbar to the upper-left corner of the grid. A new section is created.
6. Click the

   Increase Height button in the toolbar to make make this section 4 cells high.
7. In the **Properties** view below the grid, in the **Name and Definition** option group, set the following values:
   - **Name**: *CustomerDetails*
   - **Label**: *Customer Details*
8. Drag the

   Add Form Section button from the toolbar under the previous section. A new section is created.
9. Click the

   Decrease Height button in the toolbar to make make this section 2 cells high.
10. In the **Properties** view below the grid, in the **Name and Definition** option group, set the following values:
    - **Name**: *CustomerAddresses*
    - **Label**: *Addresses*

11. Drag the

    Add Form Section button from the toolbar at the right of the *Customer Details* section. A new section is created.
12. Click the

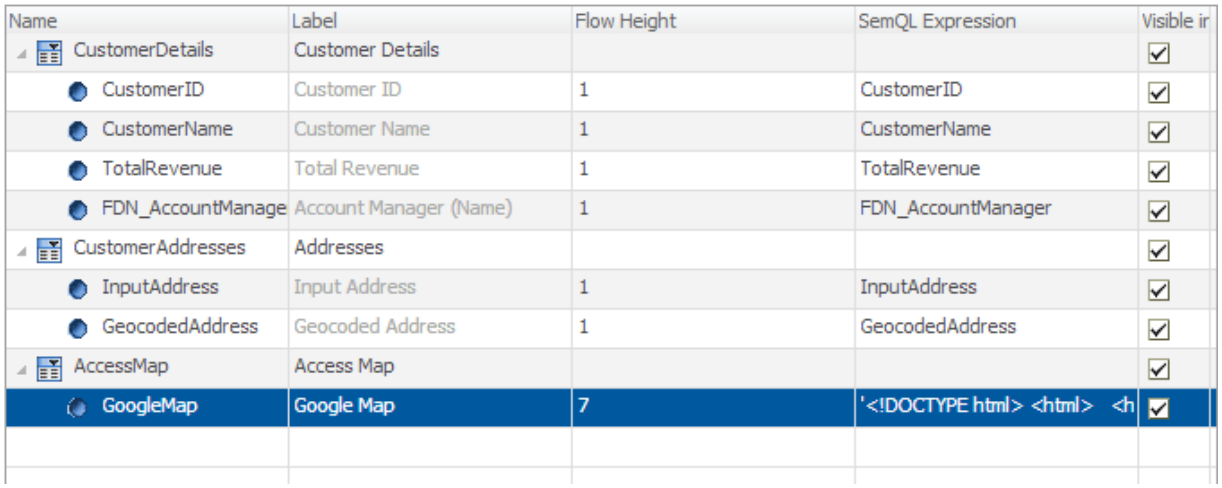
    

    Increase Height button in the toolbar to make make this section 7 cells high.
13. In the **Properties** view below the grid, in the **Name and Definition** option group, set the following values:
    - **Name**: *AccessMap*
    - **Label**: *Access Map*
14. In the **Properties** view below the grid, in the **Grid Layout** option group, deselect the **Display Label** option. The title of this section disappears in the grid.
15. Drag and drop the *CustomerID* attribute from the **Attributes** list to *Customer Details* section in the grid at the top position.
16. Repeat this operation and add the *CustomerName*, *TotalRevenue* and *FDN_AccountManager* attributes to the *Customer Details* section.
17. Repeat this operation to add the *InputAddress* and *GeocodedAddress* complex attributes to the *Addresses* section.
18. Drag the

    Add Form Attribute into the *AccessMap* section (it appears now without title).
19. In the **Properties** view below the grid, in the **Name and Definition** option group, set the following values:
    - **Name**: *GoogleMap*
    - Select the **Use Custom Label** option.
    - **Label**: *Google Map*.
    - **SemQL Expression**: Paste the code provided below.
20. In the **Grid Layout** option group, set the **Label Position** option to *Hidden*. The label of the attribute disappears.
21. In the **Display Properties** option group, set the **Display Type** to *Embedded Content*.
22. Resize and position this attribute to occupy all the space of the section. The grid should appear as shown below:

23. Select the **Flow** tab at the bottom of the editor.
24. Select the *InputAddress* attribute in the *CustomerAddresses* section. In the **Properties** view, in the **Flow Layout** option group, select the **Expandable** and **Expanded by Default** options.
25. Select the *GeocodedAddress* attribute in the *CustomerAddresses* section. In the **Properties** view, in the **Flow Layout** option group, select the **Expandable** option.
26. Select the *GoogleMap* attribute in the *AccessMap* section. In the **Properties** view, in the **Flow Layout** option group, set the **Flow Height** value to *7*. The flow should appear as shown below:



Name	Label	Flow Height	SemQL Expression	Visible ir
CustomerDetails	Customer Details			☑
CustomerID	Customer ID	1	CustomerID	☑
CustomerName	Customer Name	1	CustomerName	☑
TotalRevenue	Total Revenue	1	TotalRevenue	☑
FDN_AccountManage	Account Manager (Name)	1	FDN_AccountManager	☑
CustomerAddresses	Addresses			☑
InputAddress	Input Address	1	InputAddress	☑
GeocodedAddress	Geocoded Address	1	GeocodedAddress	☑
AccessMap	Access Map			☑
GoogleMap	Google Map	7	'<!DOCTYPE html> <html> <h	☑

27. Press **CTRL+S** to save the form view and then close the editor.

Google Map Attribute SemQL Expression

Copy and paste the code below in the SemQL expression of the Google Map attribute. It generates HTML and Javascript text to display a Map based on the InputAddress attribute.

```
'<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <script src="https://maps.googleapis.com/maps/api/js?sensor=false"></
script>
    <script>

var address= "' || InputAddress.Address || ' ' || InputAddress.PostalCode || '
 ' || InputAddress.City || '";
var zoom = 18;
var mapType = google.maps.MapTypeId.ROADMAP;
var useMarker = true;
var map;

function initialize() {
 var geocoder = new google.maps.Geocoder();
 geocoder.geocode( { "address": address}, function(results, status) {
```

```
  if (status == google.maps.GeocoderStatus.OK)
 { displayMap(results[0].geometry.location); }
 });
 window.onresize = resize;
}

function displayMap(latlng) {
 var mapOptions = { zoom: zoom, center: latlng, mapTypeId: mapType }
 map = new google.maps.Map(document.getElementById("map_canvas"), mapOptions);
 if (useMarker) {
  var marker = new google.maps.Marker({ map: map, position: latlng});
 }
 resize("");
}

function resize(e) {
 var center = map.getCenter();
 map.getDiv().style.height = window.innerHeight +"px";
 map.getDiv().style.width = window.innerWidth +"px";
 google.maps.event.trigger(map, ''resize'');
 map.setCenter(center);
}

google.maps.event.addDomListener(window, "load", initialize);
    </script>
  </head>
  <body style="margin:0px;">
    <div id="map_canvas" style="margin:0px;"></div>
  </body>
</html>'
```

To create the *DataEntryFormView* form view:

1. In the **Model Edition** view, select the **Entities > Customer > Form Views**.
2. Right-click and select **Add Form View**. The **Create New Form View** wizard opens.
3. Set the following value in the wizard:
   - **Name**: *DataEntryFormView*.
   - **Default Layout**: *Flow Layout*.
4. Click **Finish**. The **DefaultFormView** editor opens on the **Flow** tab.
5. Drag the
   
   Add Form Section button from the toolbar to the table. A new section is created.
6. Select this section. In the **Properties** view, in the **Name and Definition** option group, set the following values:
   - **Name**: *CustomerDetails*
   - **Label**: *Customer Details*

7. Drag the

   Add Form Section button from the toolbar to the table under the *Customer Details* section. A new section is created.
8. Select this section. In the **Properties** view, in the **Name and Definition** option group, set the following values:
   - **Name**: *Address*
   - **Label**: *Address*
9. Press and hold the **CTRL** key to perform a multiple selection and click in the **Attributes** list the following attributes: *CustomerID*, *CustomerName*, *TotalRevenue* and *FDN_AccountManager*.
10. Drag and drop the selected attributes into the *CustomerDetails* section. They are added to the section in their order of selection.
11. Drag and drop the *InputAddress* attribute from the **Attributes** list into the *Address* section. It is added to the section.
12. Select this form attribute. In the **Properties** view, in the **Flow Layout** option group, select the **Expandable** and **Expanded by Default** options.The flow should appear as shown below:

Name	Label	Flow Height	SemQL Expression	Visible ir
CustomerDetails	Customer Details			☑
CustomerID	Customer ID	1	CustomerID	☑
CustomerName	Customer Name	1	CustomerName	☑
TotalRevenue	Total Revenue	1	TotalRevenue	☑
FDN_AccountManage	Account Manager (Name)	1	FDN_AccountManager	☑
Address	Address			☑
InputAddress	Input Address	1	InputAddress	☑

13. Press **CTRL+S** to save the form view and then close the editor.

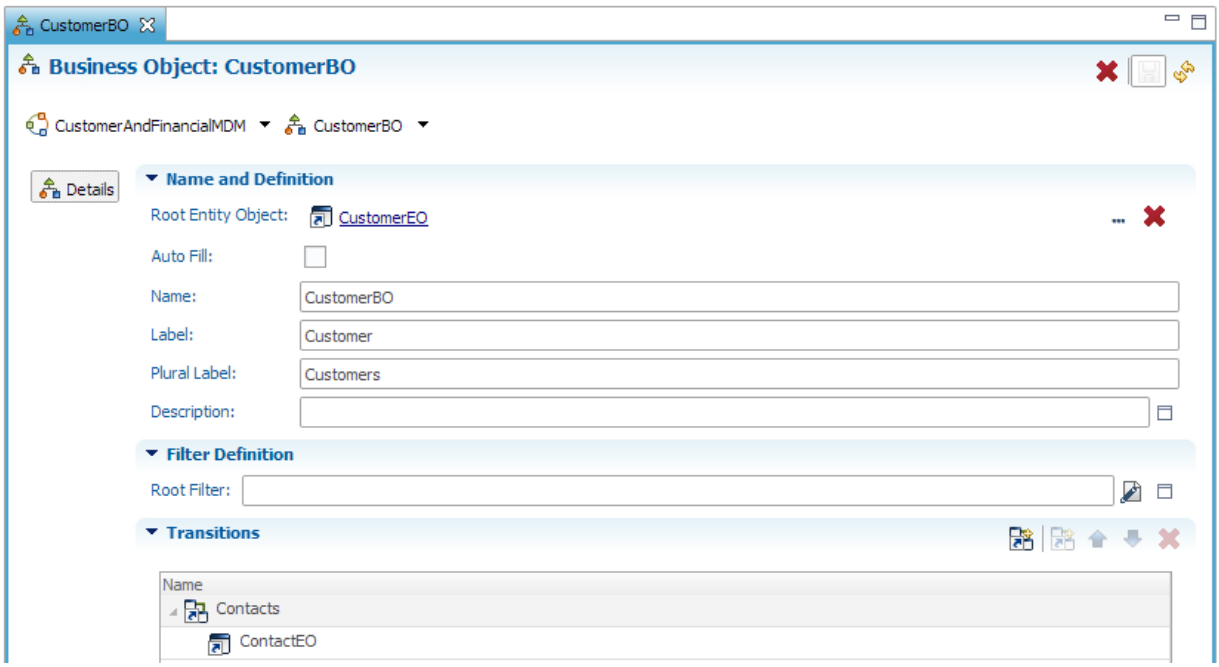
Creating Business Objects and Business Object Views

A **Business Object** assembles several entities using their relations to create an object that talks to the business. For example, the “Hierarchy of Employees” or the “Customers with their Contacts” are business objects.
In an application, **Business Object Views** are created and display business objects with specific table and form views.
In this section, we will create:
- Two business objects representing the customers and their contacts (*CustomerBO*) and the customers by account manager (*CustomersByEmployeeBO*).
- Business object views to display these business objects and to edit the customers and contacts records.

Creating Business Objects

The first business object (*CustomerBO*) groups customers with their contacts through the *ContactBelongsToCustomer* relation.

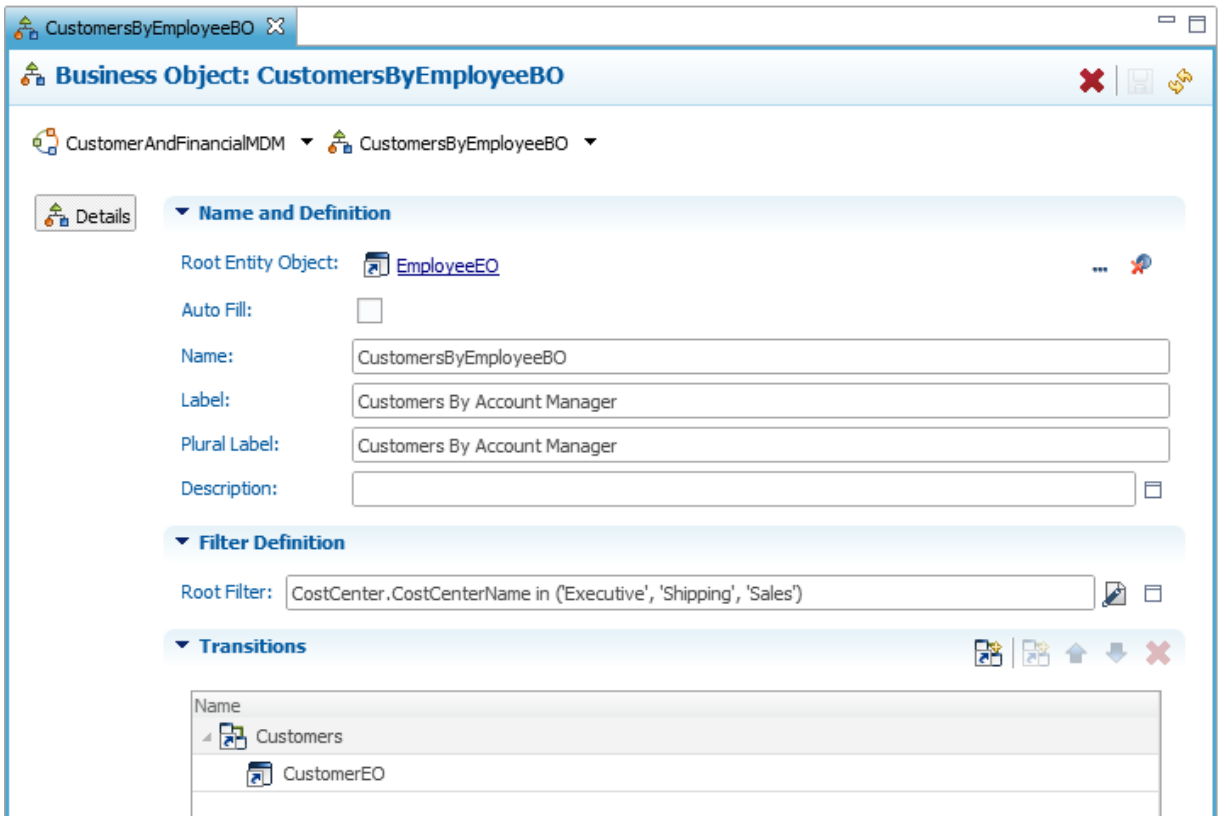
1. In the **Model Edition** view, right-click the  **Business Objects** node and select **Add Business Object...**. The **Create New Business Object** wizard opens.
   - In the **Entity** select *Customer [Entity]*
   - Click **Next**
2. Enter the following values:
   - **Name**: *CustomerBO*
   - **Label**: *Customer*
   - **Plural Label**: *Customers*
3. Click **Finish** to close the wizard. The **Business Object: CustomerBO** editor opens.
4. In the editor scroll down to the **Transitions** table.
5. Click the  **Add Transition** button. The **Create New Business Object Transition** wizard opens.
6. In the **Reference**, select *ContactBelongsToCustomer [Reference Relationship]*
7. Click **Next**.
8. Select **New Entity Object** and then click **Next**.
9. Click **Finish**. The transition is added to the **Transitions** table.



10. Press **CTRL+S** to save the editor and then close it.

Another business object *CustomersByEmployeeBO* groups the account managers with their managed customers. Account managers are in the cost centers named ‘Executive’, ‘Shipping’ and ‘Sales’.

1. In the **Model Edition** view, right-click the  **Business Objects** node and select **Add Business Object...**. The **Create New Business Object** wizard opens.
   - In the **Entity** select *Employee [Entity]*
   - Click **Next**
2. Enter the following values:
   - **Name**: *CustomersByEmployeeBO*
   - **Label**: *Customers By Account Manager*
3. Click **Next**.
4. In the **Root Filter**, enter `CostCenter.CostCenterName in ('Executive', 'Shipping', 'Sales')`
5. Click **Finish** to close the wizard. The **Business Object: CustomersByEmployeeBO** editor opens.
6. In the editor scroll down to the **Transitions** table.
7. Click the  **Add Transition** button. The **Create New Business Object Transition** wizard opens.
8. In the **Reference**, select *CustomerHasAccountManager [Reference Relationship]*
9. Click **Next**
10. Select **New Entity Object** and then click **Next**.
11. Click **Finish**. The transition is added to the **Transitions** table.



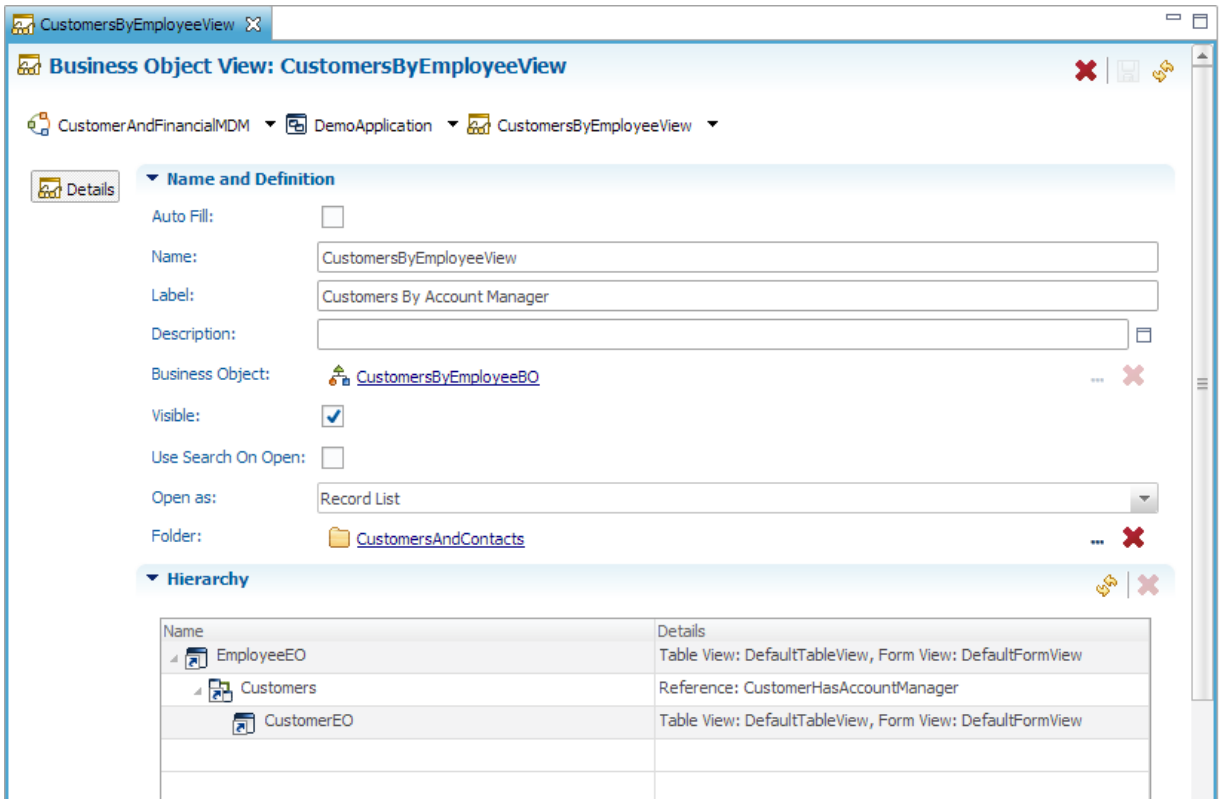
12. Press **CTRL+S** to save the editor and then close it.

Creating Business Object Views

The newly created business objects will be used in the *DemoApplication*.
To use them, several business object views must be created to define how these business objects will display. Business object views associate form and table views to business objects.

The *CustomersByEmployeeView* is a view to display the *CustomersByEmployeeBO* business object: The employees and their managed customers.

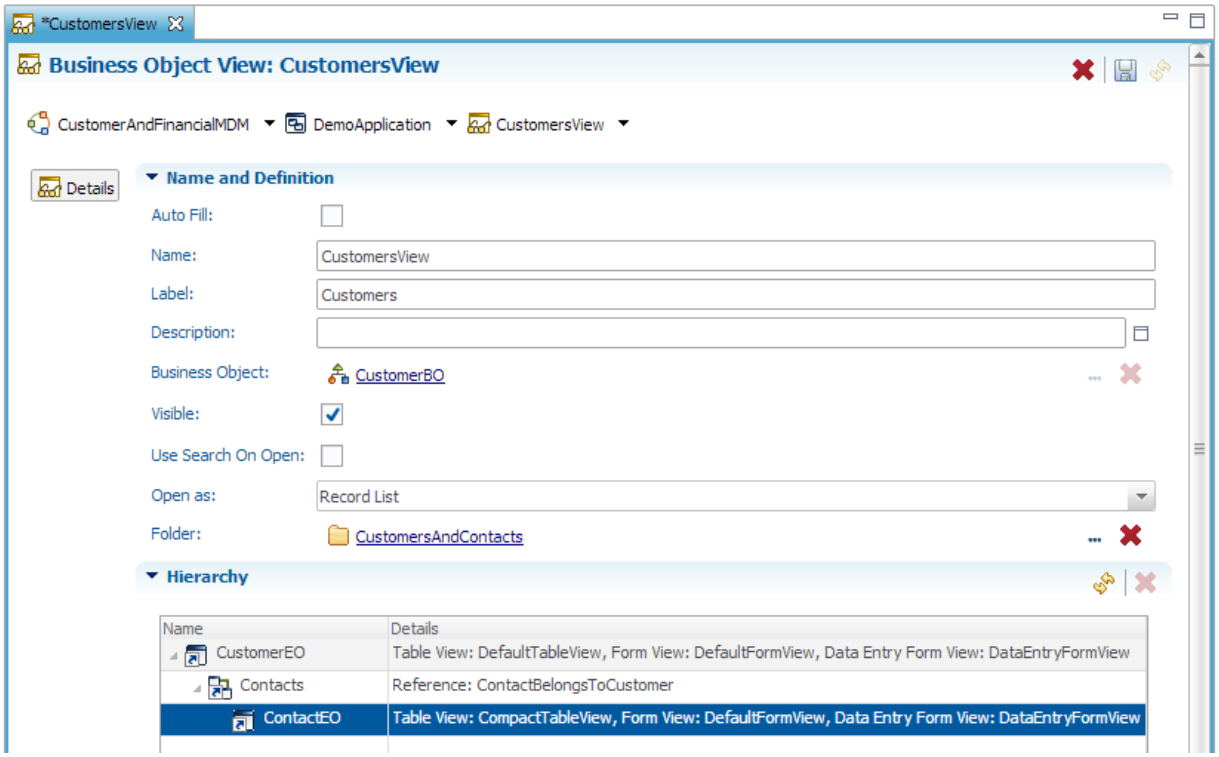
1. In the **Model Edition** view, expand the **Applications > DemoApplication > Folders** node.
2. Select the **CustomersAndContacts** folder, right-click and select **Add Business Object View...**. The **Create New Business Object View** wizard opens.
3. In the **Create New Business Object View** wizard, enter the following values:
   - **Name**: *CustomersByEmployeeView*
   - **Label**: *Customers By Account Manager*
   - In the **Business Object**, select *CustomersByEmployeeBO [Business Object]*.
4. Click **Finish** to close the wizard. The **Business Object View** editor opens.
5. Scroll down to the **Hierarchy** table, and expand the nodes to review the form/table views used for the various entities involved in the business object that this view displays.



6. Press **CTRL+S** to save the editor and then close it.

The *CustomersView* is a view used to display and author the customers and their attached contacts, based on the *CustomerBO* business object.

1. In the **Model Edition** view, expand the **Applications > DemoApplication > Folders** node.
2. Select the **CustomersAndContacts** folder, right-click and select **Add Business Object View...**. The **Create New Business Object View** wizard opens.
3. In the **Create New Business Object View** wizard, enter the following values:
   - **Name**: *CustomersView*
   - **Label**: *Customers*
   - In the **Business Object**, select *CustomerBO [Business Object]*.
4. Click **Finish** to close the wizard. The **Business Object View** editor opens.
5. Scroll down to the **Hierarchy** table, and expand the nodes to review the form/table views used for the various entities involved in the business object that this view displays.
6. Select the line for **CustomerEO** in the **Hierarchy**.
7. In the **Properties** view, in the **Name and Definition** option group, click the **... Select a Value** button for the **Data Entry Form View**. A dialog opens to select one of the form views of the *Customer* entity.
8. Select the *DataEntryFormView* and then click **OK**.
9. Select the line for **ContactEO** in the **Hierarchy**.
10. In the **Properties** view, in the **Name and Definition** option group, click the **... Select a Value** button for the **Data Entry Form View**. A dialog opens to select one of the form views of the *Contact* entity.
11. Select the *DataEntryFormView* and then click **OK**.
12. Click the **... Select a Value** button for the **Table View**. A dialog opens to select one of the table views of the *Contact* entity.
13. Select the *CompactTableView* and then click **OK**. The editor looks as shown below:



14. Press **CTRL+S** to save the editor and then close it.

Good Job! You have designed the various elements for viewing (and editing) customers.

Creating Human Workflows

Human Workflows enable business users to manage the data in the MDM hub via an application. In this section, we will configure two workflows to enter customers/contacts and to manage customer duplicates.

When users want to manage the master data, they initiate an **Activity** based on a human workflow. This activity follows the workflow through **Transitions** and **Tasks** which are assigned to roles, claimed, processed and then completed by users. The last task of a workflow can submit (or cancel) the data changes done in the activity, and start a data certification process with these changes.

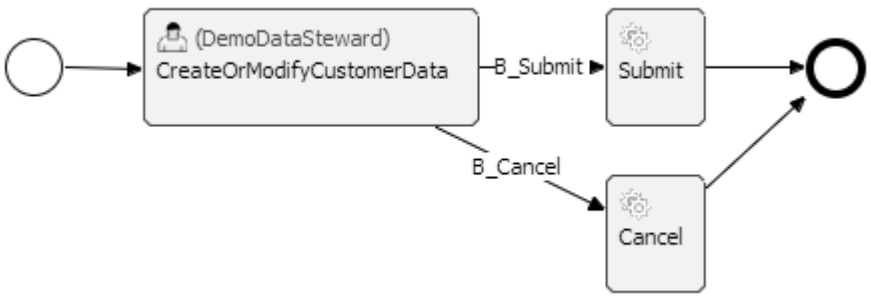
There are two types of human workflows in Semarchy Convergence for MDM:

- **Data Entry Workflows**: These workflows allow data stewards or business users to contribute to the hub as manual publishers. The data entered via these workflows goes through the data certification process to create golden data.
- **Duplicate Management Workflows**: These workflows allow data stewards to override the decisions taken by the matchers running in the hub. Through these workflows, stewards can either manually match unmatched records, or split duplicate groups that were incorrectly matched (false matches).

Creating a Data Entry Workflow

Authoring data for the new *Customer* entity will be performed through a new data entry workflow called *CustomersCreationProcess*.
This single-task workflow involves only users with the *DemoDataSteward* role.

1. In the **Model Edition** view, expand the **Applications > DemoApplication > Human Workflows** node.
2. Right-click the **Human Workflows** node and select **Add Data Entry Workflow...**. The **Create New Data Entry Workflow** wizard opens.
3. In the **Create New Data Entry Workflow** wizard, enter or select the following values:
   - **Name**: *CustomersCreationProcess*
   - **Label**: *Edit Customers*
   - **Data Entry Publisher**: *DataEntry [Publisher]*
   - **On Submit Job**: *DE_CUSTOMER_CONTACT [Job]*
   - **Initiator Role**: *DemoDataSteward*
4. Click **Next**. This step shows the various actions allowed in this workflow: *Checkout* actions allow editing existing records or fixing rejects. *Create New Record* allows adding records via the workflow. Leave all the actions selected.
5. Click **Next**.
6. In the **Available BO Views** list, double-click the *CustomersView* to add it to the **Selected BO Views**.
7. Click **Finish** to close the wizard. The workflow editor opens on the **Diagram** tab. The workflow was automatically created with a simple task.
8. In the workflow diagram, select the *Task1* task.
9. In the **Properties** view, in the **Name and Definition** option group, set the following values:
   - **Name**: *CreateOrModifyCustomerData*
   - **Label**: *Create / Modify Customer Data*
   - **Assigned to Role**: *DemoDataSteward*
10. The workflow diagram appears as shown below.



11. Press **CTRL+S** to save the editor and leave it open.

Data entry workflows support interactivity through enrichers and data quality validations triggered while entering data or moving the workflow to a new task.
We will configure all validations and enrichers to run when the focus leaves the data entry fields, except one email enricher that will be triggered manually. We will also configure validations to prevent users from submitting data that does not comply with certain validations, as well as warn them of errors before submission.

Configure the Task:

1. Select the **Properties** view (lower panel of the workbench).
2. Select the **CreateOrModifyCustomerData** task in the workflow.
3. In the **Properties** view, select the **Enrichers** tab. This tab shows the list of enrichers triggered while entering data in this task.
4. Select all the lines in the enrichers list, right-click and then select **Field Exit**.
5. Now select the *GeocodeInputAddress* and *StandardizeContactEmail* enrichers in the list, right-click and then select **Manual**.
6. Select the **Validations** tab in the **Properties** view.
7. Select all the lines in the validations list, right-click and then select **Field Exit**.
8. Press **CTRL+S** to save the editor and leave it open.

Configure the Transition:

1. In the workflow, select the **B_Submit** transition that links the *CreateOrModifyCustomerData* task to the *Submit* task.
2. Select the **Enrichers** tab in the **Properties** view.
3. Select all the lines in the enrichers list, right-click and then select **Execute Enricher**.
4. Select the **Validations** tab in the **Properties** view.
5. Select all the lines in the validations list, right-click and then select **Block**.
6. Now select the *Account Manager – Reference* validation in the list, right-click and then select **Warn**.
7. Press **CTRL+S** to save the editor and then close it.

Creating a Duplicate Management Workflow

We will now create a *CustomerDuplicatesProcess* workflow which manages the customer duplicates.

1. In the **Model Edition** view, expand the **Applications > DemoApplication > Human Workflows** node.
2. Right-click the **Human Workflows** node and select **Add Duplicates Management Workflow...**. The **Create Duplicates Management Workflow** wizard opens.
3. In the **Create New Duplicates Management Workflow** wizard, enter or select the following values:
   - **Name**: *CustomerDuplicatesProcess*
   - **Label**: *Manage Duplicate Customers*
   - **Managed Entity**: *Customer [Entity]*
   - **On Submit Job**: *DE_CUSTOMER_CONTACT [Job]*
   - **Initiator Role**: *DemoDataSteward*
4. Click **Finish** to close the wizard. The workflow editor opens on the **Diagram** tab. The workflow was automatically created with a simple task.
5. In the workflow diagram, select the *Task1* task.
6. In the **Properties** view, in the **Name and Definition** option group, set the following values:
   - **Name**: *VerifyCustomerDups*
   - **Label**: *Verify and Validate Customer Duplicates*
   - **Assigned to Role**: *DemoDataSteward*
7. Press **CTRL+S** to save the editor and then close it.

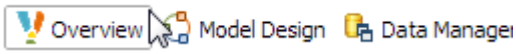
Congratulations! Your first application is ready to go live!

Using the Application

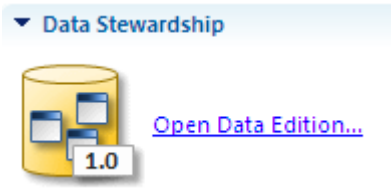
In an **Application**, data stewards and business users can browse the MDM hub content using the **Business Object Views**. They can also author (create or modify) master data in the hub and manage duplicates using the **Human Workflows**.

Connecting to the Application

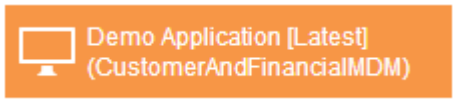
1. In the top-right toolbar, select the **Overview** perspective.



2. In the **Overview** screen, press the CTRL key and then click the **Open Data Edition** link under the **Data Stewardship** group to open it in a new tab.

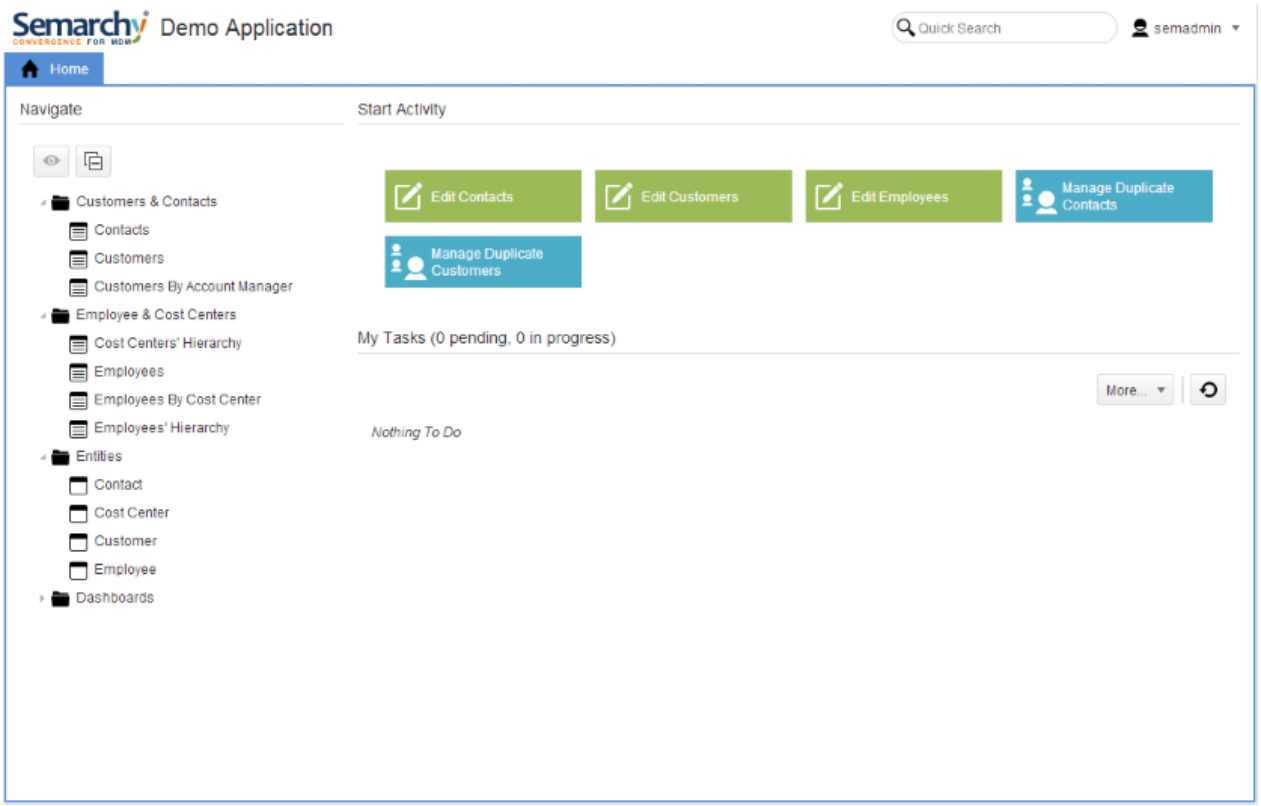


3. A new tab opens, showing the applications available. In the **Available Applications** section click the **Demo Application [Latest] (CustomerAndFinancialMDM)** button to open this application. The application opens.



Note: At design-time, the application is generated every time you access it (the startup time is in fact the application generation time). In production, it is generated once.

The Demo Application



The application **Home** page provides an overview of the hub:

- The **Navigate** view shows the business object views classified in their folders, as well as the list of the entities in the hub. Double-clicking on a business object view opens a new editor showing the data.
- The **My Tasks** view is a summary of the human workflows in progress.
- The **Start Activity** view contains shortcuts to start new workflows.

> **Note**: Access to the data is secured by privileges. As the data steward, you can access all the entities, but other users have a more limited access as defined in the model security, which is not covered in this tutorial.

Navigating the Master Data

In this section, we take the role of a business user who wants to browse the customer data.

Browsing

Let's look for a customer record called *GADGETRON*.

1. Double-click the **Customers** view in the **Customers and Contacts** folder. A new editor opens showing the list of customer data certified in the hub in the **Hierarchy** view and in a table view.
2. In the editor toolbar, click the
   
   Display Settings button and then select **Show Lineage** to disable the lineage.
3. Click the **Customer Name** column header on the table to sort records by customer name.
4. Use the
   
   Next Page and
   
   Previous Page buttons to move from page to page until you see *GADGETRON*.

Quick Search

Another way to access GADGETRON is using a quick search.

1. Click the *Quick Search* field in the application header.
2. Enter `Gad` in this field, and then press ENTER.

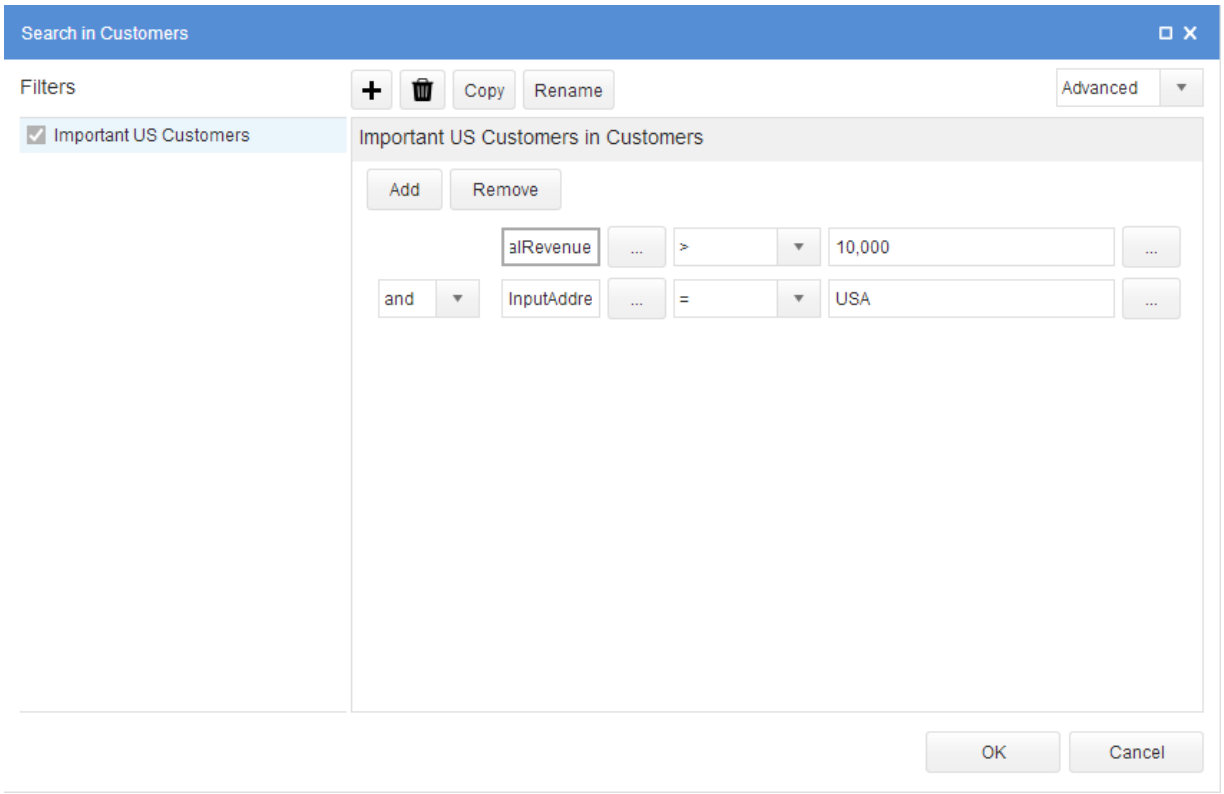
   

   A dialog opens, asking you the business object view to search into.
3. Select the **Customers** business object view and then click **OK**. The **Customers** editor opens with two records matching the search.

Advanced Search

Let's try some advanced filtering. We will search for "US customers with revenue greater than 10,000" and then export them.

1. Click the
   
   Search button in the editor toolbar to open the Search dialog.
2. The current <*Quick Search*> filter is displayed with the current full-text search pattern: `%Gad%`.
3. Select **Advanced** in the search type dropdown box, in the upper right corner of the dialog.
4. Click the **...** (**Select Attribute**) button.
5. In the **Select Attribute** dialog that opens, select *TotalRevenue* and then click **OK**.
6. Select the "greater than" (>) operator.
7. In the value field, enter `10000`.
8. Click the **Add** button.
9. Select the **...** (**Select Attribute**) button. In the **Select Attribute** dialog that opens, expand the *InputAddress* complex field, select *Country* under that node and then click **OK**.
10. Click the **...** selector button to select the country USA.
11. Double-click on the filter name to rename the <*Quick Search*> filter to *Important US Customers*.



12. Click **OK**. The filter is saved, applied and only US customers with total revenue greater than 10,000 appear.

Export

1. Select **More... > Export** in the editor toolbar.
2. Select the suitable **Export Format** (Excel or CSV).
3. Click the **OK** button to download the data file and open it with the appropriate editor.

Browsing References

1. In the list of customers, use the
   »
   Next Page and
   «
   Previous Page buttons to move from page to find the *GADGETRON* record.
2. Click this record in the list. A page appears showing the details for this record. The **Contacts** tab shows the list of all contacts attached to this customer.
3. In the hierarchy, expand the *GADGETRON > Contacts* node to view the contacts attached to this customer.
4. Click the *Britney Bell* contact to open its form view.

5. In this form, click the *GADGETRON* link in the *Customer Details* section to go back to the parent customer record.
6. In the *Gadgetron* record, click the *Matthew Weiss* link to open the record of the account manager for this customer.
7. Return to the **Customers** editor.

Authoring Master Data

The same business user now wants to change GADGETRON's total revenue and create a new customer called SEMARCHY.

Creating and Modifying Records

1. In the *GADGETRON* editor toolbar click the
   
   Edit the Current Selection button. A new activity automatically starts and an editor opens to edit *GADGETRON* in this activity.
2. Change the **Total Revenue** value to `145000` and then click **Save**. You return to the **Edit Customer** activity editor.
3. Select the
   
   Create New Customer button in the toolbar. An editor opens to create a new customer.
4. Enter the following values:
   - **Customer Name**: *SEMARCHY*
   - **Total Revenue**: 6946000
   - **Account Manager**: Click the
     
     More... button. The list of employees opens. Click *Steven King* in this list. The list closes and this employee is set as the account manager.
5. In the *Address* section, enter:
   - **Address**: *750 Menlo Ave #250*
   - **Postal Code**: *CA 94025*
   - **City**: *Menlo Park*
   - **Country**: *USA*
   - Click the **Save** button to save this entry.
6. Click the **Complete** button in the editor toolbar and then select **Submit** to complete this task.
7. Click **OK** in the **Complete Task** dialog, . The data is validated and published to the hub.

Note: As you enter data in the forms, red markers indicate the various errors detected. If you move your mouse cursor on these indicators, the details of the error display in a tooltip. Data quality rules are enforced as part of the workflow, and you cannot submit incorrect data.

Reviewing the Changes

Perform a quick search for the `SEM` and `GADGE` customers to see the new and updated customer records:

- A new customer called SEMARCHY must appear now.

- GADGETRON must now have a Total Revenue of 145,000.

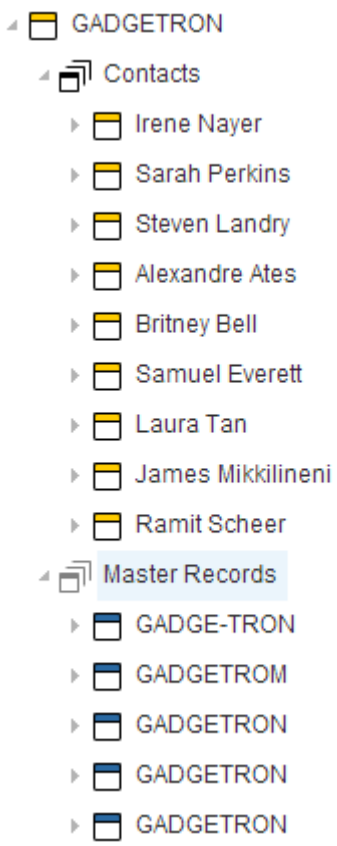
Data Stewardship

Data stewards are responsible for maintaining the data in the hub. In addition to the navigation and authoring operations, they use data lineage, duplicate management and rejects detection for governing the data.

Using the Lineage

The **Lineage** reflects the process that led to the master records and allows drilling down into the various stages of this process.

1. Select the **Home** tab.
2. Double-click the **Customers** view in the **Customers and Contacts** folder. A new editor opens showing the list of customers.
3. In the editor toolbar, click the
   
   Display Settings button and then select **Show Lineage** to enable the lineage.
4. Sort the records by customer name and then use the
   
   Next Page and
   
   Previous Page buttons to move from page to page until you see *GADGETRON*.
5. Expand the node corresponding to a GADGETRON golden record in the **Hierarchy**.
6. Expand the **Master Records** node. The list contains the master records that led to the consolidated golden record.



7. Click the **Master Records** node under the GADGETRON record in the **Hierarchy**. The list of master records appears.

GADGETRON

Contacts | Master Records

Customer ID	Customer ID (Golden ID)	Customer Name	Total Revenue	Account Manager (Name)	Input Address	G
MKT.1260	2	GADGE-TRON	4,770	Matthew Weiss	710 KAPIOLANI BV HONOLULU USA	
CRM.1251	2	GADGETROM	1,405	Matthew Weiss	711 KAPIOLANI BLVD HONOLULU USA	
CRM.12019	2	GADGETRON	2,558	Matthew Weiss	711 KAPIOLANI BOULEVARD HONOLULU USA	
CRM.X3452	2	GADGETRON	14,500	Lex De Haan	711 KAPIOLANI BLVD HONOLULU USA	
DE.38b81371-f66b-	2	GADGETRON	145,000	Matthew Weiss	711 KAPIOLANI BOULEVARD HONOLULU USA	

You can review the following:

- The records displayed are duplicates coming from the CRM and MKT (Marketing) applications, plus one authored earlier in this tutorial. These have been matched by the Matcher created in this tutorial.
- During the consolidation phase, the highlighted values have been taken to create the golden record as defined by the consolidator.

- The *Customer Name* and the *Input Address* have been transformed to uppercase as per the SemQL Enricher.
- If you have activated the Plug-in Enricher, the *Geocoded Address* is loaded. Otherwise it is empty.

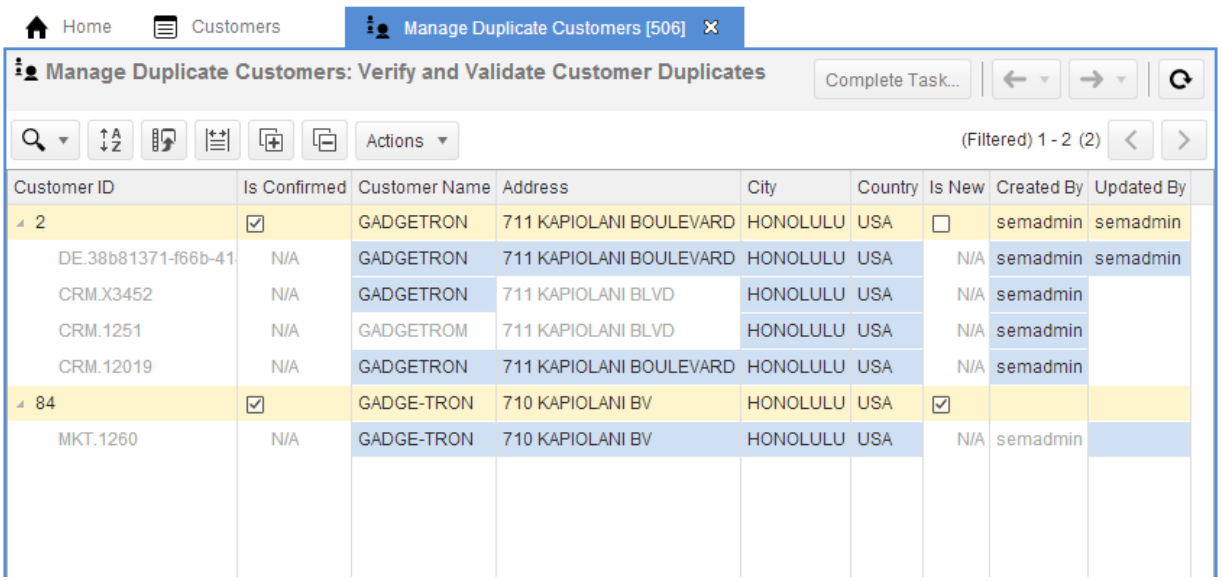
Managing Duplicates

When the automated duplicates detection does not provide appropriate results, it is possible to split or merge duplicate groups.

> A number of records, published from source applications, have been automatically matched and consolidated into the GADGETRON golden record. One of them was incorrectly merged. In reality, *GADGE-TRON* is in fact a different legal entity than *GADGETRON*. They share the same address, and the matching process interpreted the name and address similarity for a match. The data steward must fix this.

Un-Merging Duplicates

1. In the **Master Records** table for *GADGETRON*, check the selection box for the *GADGE-TRON* master record (Customer ID = MKT.1260).
2. In the toolbar, click the
   
   Duplicates Management button and then select **Move to New Golden**.
3. A new **Manage Duplicate Customers** workflow opens. In the editor, the two records resulting from the un-merge operation appear.



4. Click the **Complete Task** button in the editor toolbar to complete this task.
5. In the **Complete Task** dialog, make sure that the **Action to Perform** is set to *Submit* and then click **Finish**. The data is validated and published to the hub.

Merging Duplicates

We can revert this operation just as easily.

1. Make a quick search for GADGE for **Customers**. The GADGETRON and GADGE-TRON records that we have unmerged appear.
2. check the selection box for both.
3. In the toolbar, click the
   
   Duplicates Management button and then select **Merge**.
4. A new **Manage Duplicate Customers** workflow opens. In the editor, the record resulting from the merge operation is shown.
5. Click the **Complete Task** button in the editor toolbar to complete this task.
6. Click **Finish**.

Note: The duplicate management process allows confirming, splitting or merging detected duplicates. Note that the decisions taken by the steward are preserved in future loads and override any automated duplicate detection.

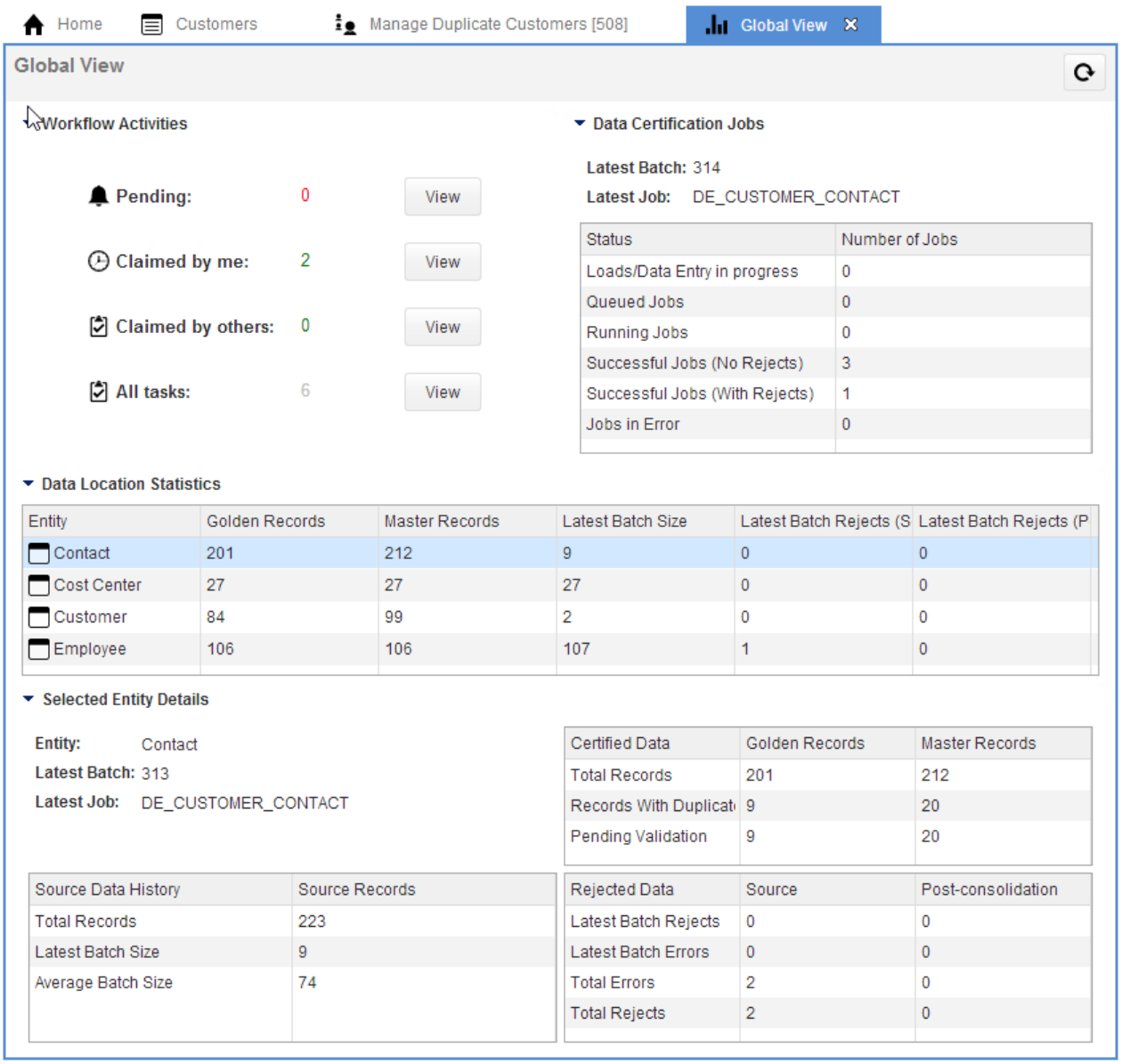
Using the Home Dashboard

The various **Dashboards** provide metrics for the hub.

1. Select the **Home** tab.
2. Select **Dashboards > Global View** in the **Navigate** section. The **Global View** page opens.

This dashboard displays several sections:

- **Workflow Activities** displays a summary of the activities in progress.
- **Data Certification Jobs** shows the status of the certification jobs in progress, queued or completed.
- **Data Location Statistics** shows the aggregated metrics for all the entities of the MDM hub. If you select one entity in this list (for example: *Contact*), the detailed metrics for this entity are displayed in the **Selected Entity Details** section of the dashboard.



Tip!: More dashboards are available at the same location when Convergence Pulse is configured to run with your instance of Convergence for MDM.

Congratulations! As a data steward, you have successfully reviewed duplicates, managed false duplicates, monitored workflows, certification jobs and the hub's statistics in the dashboards!

Summary

Summary

> Congratulations! You have now completed your first MDM project with Semarchy Convergence for MDM.

In this tutorial, you learned how to:

- Install Convergence for MDM,
- Design a model, including customized data types, entities, integration artifacts (enrichers, constraints, matchers and consolidators),
- Deploy the Model in an MDM Hub,
- Load the MDM Hub and automatically run integration jobs,
- Create an Application for business users and data stewards to access and perform changes in the MDM Hub,
- Use this application to orchestrate business processes and to navigate through your golden and master data,
- Review the golden data certified by Convergence for MDM and drill through the data lineage.

Going Further with Convergence for MDM

You have learned how to use Convergence for MDM for a typical MDM project. But Convergence for MDM is capable of addressing any type of master data projects in many functional areas, including Products, Customer, Parties, Cost Centers, etc.

Convergence for MDM includes the following features to support your MDM initiatives:

- **Comprehensive Logical Modeling** for faster development and optimal reactivity to changes at a lower cost.
- **Data and Metadata Versioning** to increase your confidence for data compliance regulations and supporting what-if analyses.
- **Non-Intrusive Convergence Hub** allowing applications to seamlessly push data to Convergence for MDM and consume certified data.
- **Generating Integration and Certification Processes** guaranteeing fully certified data.
- **Leveraging the Power of Database Engines** for high performance and maximal scalability for master data certification and access.
- **Web 2.0 and Cloud-Ready Solution** providing the lowest cost of deployment on the market.

Learn More

> **Note**: The **Semarchy Convergence Documentation Library**, including the development, administration and installation guides is available online at the following URL: http://semarchy.com/download/

In addition to the product manuals, Semarchy provides other resources including whitepapers, datasheets, and a complete set of videos demonstrating the product features.

The resources are available on the Semarchy Website.