**Master Data Management in SQL Server 2012: Use case of managing Data warehouse Dimensions with Master Data Services**

**Introduction**

While raw fact data and dimension business keys are relatively easy to acquire, the real challenge of a Datawarehouse project is to get Master Data or so-called categorical-type dimension attributes correct. This is because in many cases the categorical attributes are not required by the transactional systems to properly book a transaction and several operational systems like ERP, CRM, SCM and HR, might not be in sync on key master data elements like Customer, Product, Employee, Cost center etc.

However those attributes are essential for analytical applications because they are used to group and aggregate data. Incorrect categorical attributes clearly impact to relevance of the resulting Business Intelligence applications.

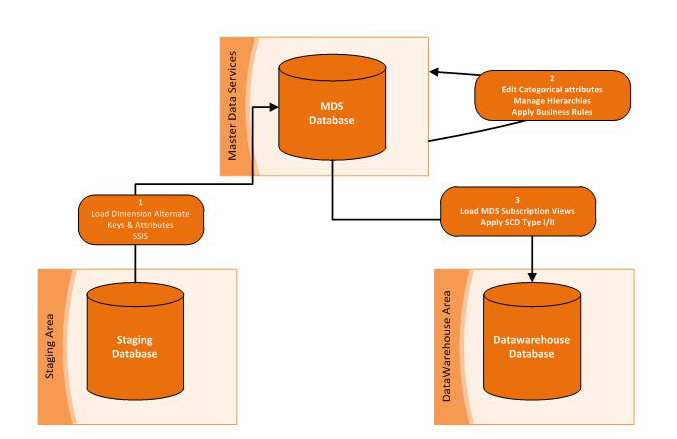
This insight discusses the problem of dimension and hierarchy management for data warehouses and multidimensional views and describes a solution to the categorical-type dimension attributes problem using Microsoft SQL Server 2012 Master Data Services (MDS).

Master Data Services is Microsoft’s answer to the Master Data Management challenge. Master Data Management (MDM) is a set of tools and processes that aim to deliver a single clean and consistent view of each master data entity (product, customer, employee, financial accounts, etc.) that exists within the organization.

Master Data Services (MDS) was already present in SQL Server2008 R2 and was the result of the integration of the acquired technology of Stratature into SQL Server. In the SQL Server 2012 "Business Intelligence” edition, Microsoft now delivers a nice, complete and reliable version 2 of this Master Data Management solution.

**High Level Process**

Before digging into the different steps that need to take place when using Master Data Services to manage a Data warehouse dimension, let’s start with an overview of the fully automated process:



*Figure 1: High Level Process using MDS to Manage your Dimension Hierarchy*

The main idea is to avoid a direct link from your operational data to your Datawarehouse by positioning Master Data Services between the two. It is within Master Data Services that the categorical-type attributes will be defined to enrich the business keys stored in the operational systems.

As a first step, the business keys and the relevant attributes of your dimension data will be loaded into the Master Data Service Application from your Staging Environment using Integration Services (SSIS). The work that needs to be performed in the staging area is exactly the same as the one you do when you are loading data directly in the dimensions of your Datawarehouse.

The next step will be to enrich the data by adding new categorical attributes and/or managing hierarchies leveraging the functionalities of Master Data Services. This task will be done by the data steward (the person that is responsible for maintaining the dimension data).

The last step will be to load dimension data from Master Data Services using the standard "subscription” view interface into the Data warehouse to apply "Slowly Changing Dimension”-processing where required and keep a snapshot of historical data.

**Designing the Object Model**

The first thing that needs to be done is to create a new model via the "System Administration” menu. The model is the highest level object in the MDS object model. It represents an MDM subject area. A model defines the structure of data in your master data management solution.

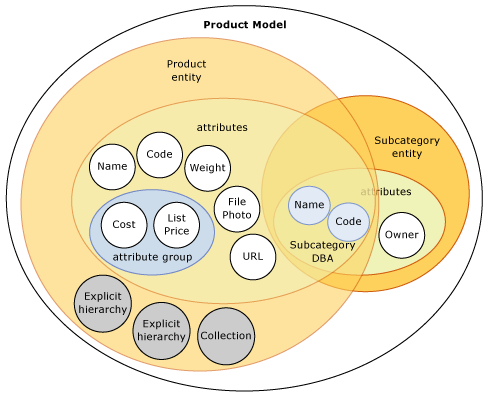


*Figure 2: Create Model - Master Data Services*

The main objects that a model contains are:

* **Entities**. They represent a collection of attributes.
* **Attributes**. Attributes are the building blocks of entities in MDS.
* **Hierarchies**. This is a tree structure that you can use to regroup similar members for reporting purposes.

Models organize the structure of your master data. Your Master Data Services implementation can have multiple models that each group similar kinds of data. For example, you can create a Product model to contain product-related data or a Customer model to contain customer-related data.

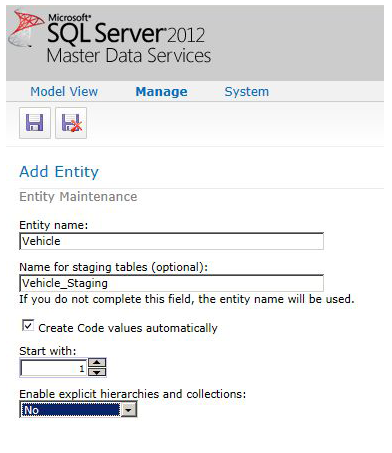


*Figure 3:*  
*Relationships among objects in one model*

**Entities**

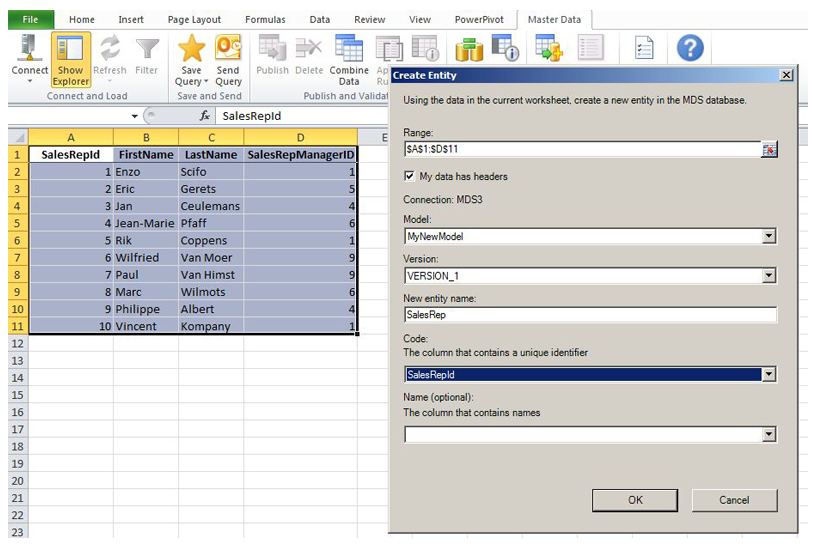
Once the model has been created, one can add entities to the model. There are two ways to do that:

* Via the "System Administration” menu. In that case, you need to define all your attributes via the web interface. If you need to define a "domain” attribute (an attribute for which the member value is limited to the values of another entity) make sure to first define the entities from which you want to derive the attribute.
* Via the Master Data Service Excel Add-In. In that case, the attributes and their respective properties are inferred from your Microsoft Excel data set. You still have the flexibility to modify the automatically generated properties of the attributes via the "attribute property” button.



*Figure 4:*  
*Create Entity from Web Interface*

[[Master Data Management in SQL Server 2012: Use case of managing Data warehouse Dimensions with Master Data Services](https://www.element61.be/sites/default/files/figure-4-master-data-services-sql-server-2012-create-entity-web-interface.png?width=500&height=500)](https://www.element61.be/sites/default/files/figure-4-master-data-services-sql-server-2012-create-entity-web-interface.png?width=500&height=500)



*Figure 5: Create Entity from XLS Add-In*

**Attributes**

For every entity, there are two attributes that are created by default:

* The "code” attribute is a unique identifier of your records. It should contain the business key of your dimension.
* The "name” attribute, which is optional.

Even though the "name” attribute is optional, it should be filled in with care because it will be the only way for the end-user to easily recognize the member of an entity in the MDS 2012 Web interface.

There are three different types of attributes:

* The "domain-based” type attribute is defined as an entity within the model. The values for this attribute type work as a constrained list based on the values of the configured entity. This attribute type is critical because it defines the domain relationships between entities which are used to set-up derived hierarchies as we will see shortly.
* The "free-form” type attribute is constrained only by its data type. The data type of this attribute can be Number, Text, DateTime or a Link. Each data type allows you to configure relevant properties such as length, decimal places and input mask.
* The "file” type attribute is used to store files, such as images or documents. An example of this attribute type would be something like an employee badge photo or product photo.

**Business Rules**

When defining the properties of the attributes of one entity, we can already ensure consistency in data type and referential integrity between the different entities via the use of the "domain attribute”. Nevertheless there might still be other requirements related to consistency and accuracy of data that we want to achieve. For that purpose, you can define business rules within Master Data Services.

Business rules are **If***Conditions***Then***Actions* statements that can be created via the Master Data Manager User interface. If an attribute value meets a specified condition, then an action is taken. Possible actions include setting a default value, changing a value, performing attribute validation, or initiating a SharePoint workflow.



*Figure 6:*  
*Add Business Rules via Business Rule Editor*

To use business rules, you must first create and publish your rules, then validate your data against the published rules. If an attribute value does not pass "business rule validation”, the value can still be saved. Validation issues are displayed in the user interface, and a version cannot be committed until all attributes pass business rule validation.

Typical examples of usage of business rules when managing dimensions could be :

* Check if a field is empty or blank and trigger a validation error
* Generate "Full Name” attribute by concatenating attribute "First Name” with attribute "Last Name”
* Derive the value of a categorical attribute from the value of another attribute

**Hierarchies**

The next step will be to define some hierarchies on top of your existing entities. A hierarchy is the organization of members into groups based on relationships between entities or specified consolidation points (Consolidated Members). It defines how you view and aggregate information. Hierarchies in Master Data Services are managed in the Hierarchy Explorer.

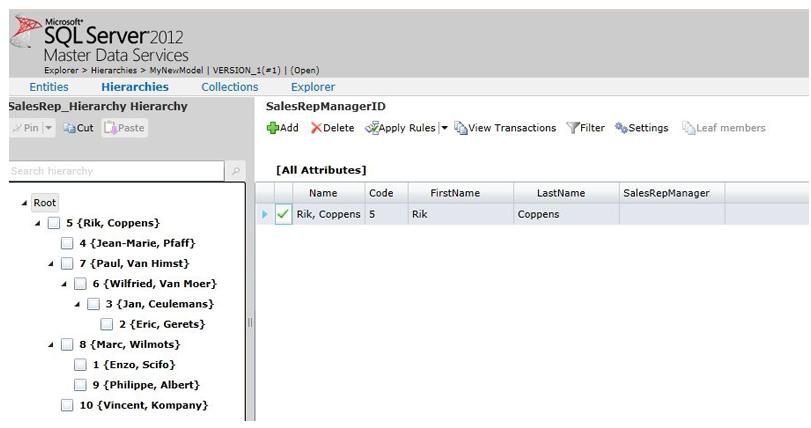
Hierarchies can be either ragged or level-based.

Ragged hierarchies have a varying number of levels throughout the hierarchy; each consolidation point is created to support specific analysis needs, but there is no predefined level structure. These types of hierarchies are flexible in allowing for the explicit definition of each consolidation point without restriction to the number of levels along any branch of the hierarchy.

Level-based hierarchies have a consistent structure. Each level in the hierarchy is defined by a single entity. All leaf members must be assigned at the same level of depth from the top – they cannot be assigned to multiple levels in the hierarchy.

There are three different hierarchies in MDS

* "Explicit hierarchies” use Consolidated Members to group other Consolidated and Leaf Members in order to make a hierarchy. The structure can be completely ragged, with each Consolidated Member having an unlimited number of Leaf or Consolidated Members beneath it.
* "Derived hierarchies” are hierarchies created from the domain-based attribute relationships between entities resulting in a fixed-depth hierarchy.
* "Recursive hierarchies” allow an Entity Member to contain an attribute that points to another Member within the same Entity. This is a classical parent-child structure. A typical example of this would be "Employee Structure”.



*Figure 7:*  
*Manage Hierarchy in Master Data Services*

Explicit hierarchies can be defined as mandatory or non-mandatory. A mandatory hierarchy in MDS contains all of the leaf-level members of the entity (once and only once) and ensures completeness.

When multiple mandatory hierarchies are defined, each leaf-level member must be included in each hierarchy. Non-mandatory hierarchies do not require inclusion of all leaf level members. Leaf members that are not included (or "used" by the hierarchy) are displayed under the "Unused" node in the hierarchy view. When using "derived hierarchies” the "mandatory” parameter does not exist but you can get the same result in using business rules to enforce the existence of the attributes and therefore making sure that your hierarchy is complete.

**Loading data into the model**

Now that the model has been created, we need to populate it with data. There are three different ways to achieve this:

* Via the Web Interface trough the "Explorer” Menu. This is particularly suited when you need to do only a small number of changes or when you need to manage hierarchies.
* Via the MS-Excel Add-in. This is particularly suited when you need to do a one-time load of a relatively large number of members. You can benefit from the productive Excel environment to prepare your data and submit them all at once.
* Via the staging tables using SQL Server Integration Services (SSIS). This is the option to use when your data are evolving every day and you want to automate the process of loading your data.

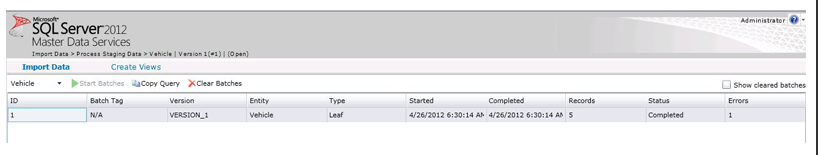
The typical scenario for a Data warehouse would be to automate the load of your business key dimension attributes via the staging tables and manage the hierarchy and the categorical attribute via the web interface.

The Master Data Services database includes three staging tables:

* stg.name\_Leaf. Create, update, and delete leaf members and their attributes.
* stg.name\_Consolidated. Create, update, and delete consolidated members and their attributes.
* stg.name\_Relationship. Move members in an explicit hierarchy.

The name of each table is specified when an entity is created and cannot be changed afterwards. Each of the staging tables contain a field "import\_type” where we can specify what type of action (Create/Update/Delete member) is required for the particular row when staged data matches data that already exists in the Master Data Services database.

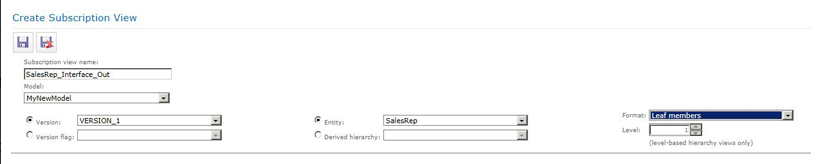
You can load your data in these tables using SSIS. Once the tables are loaded, one can initiate the staging process to load the data from the staging tables into the appropriate MDS tables via the "Interface Management” menu. This can also be fully automated it using a standard MDS stored procedure.



*Figure 8:*  
*Import Data into Master Data Services via Integration Menu*

**Subscriptions Views**

Now that we have a model with correct data in it, we want to be able to read this data and move them to our Data warehouse dimension tables. To do that, you can define "subscription” views via the "Integration Management” menu.



*Figure 9:*  
*Create Subscriptions Views in Master Data Services*

Subscription views provide a simple way to extract and publish master data from MDS using Integration Services, SQL scripts or any other tools and techniques that access a standard SQL Server view. MDS administrators may generate subscription views as needed from the "Integration Management” page of the Master Data Manager Web User Interface ("UI”).

Subscription views denormalize the master data and apply recognizable names to columns. This makes them much easier to work with than the underlying tables which are normalized for performance and data integrity.

Once generated, subscription views appear as normal views within the "mdm” schema of the MDS SQL Server database. Subscription views are best suited for bulk, batch-oriented exports of master data (typical data warehouse scenario). For other integration scenarios like event-based integration, you can use the change tracking notification feature.

**Loading Master Data in the Datawarehouse Dimension**

Some might be tempted to gain some development time by bypassing the load of the subscriptions views into the Datawarehouse. Indeed technically, a subscription view is like any table and therefore you could look it up to derive the surrogate keys of your fact table in your ETL process and you can also directly link it to your cube in Analysis Services. This is far from being a good idea for the following reasons:

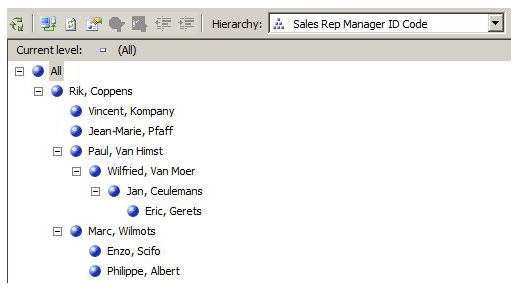
* MDS only contains the current version of your master data while your dimension should contain full history of your master data
* The list of the attributes of your master data entity may vary along time
* SCD Type II is something that can only be handled in your data warehouse
* the Data model of MDS is typically highly normalized to ensure data integrity. The subscription view that is created on top of that will never be as performant as a physical dimension table holding the same information in your data warehouse. This might create serious performance issues if you need to run reports directly on your database layer.

We strongly recommended that you always move your data from MDS into your Datawarehouse.

**Design your Dimension with Analysis Services**

To leverage the flexibility offered by "explicit” and "recursive” hierarchies to add levels into the hierarchy, you need to define your dimension in your cube as "parent-child”. Unlike regular dimensions, which are defined with a number of levels that determines the number of levels seen by end users, a parent-child dimension is defined with a single level of a special type that usually produces multiple levels seen by end users.

The number of displayed levels depends on the contents of the columns that store the member keys and the parent keys. This number can change when the master table is updated and the dimension table and the cubes using the dimension are subsequently processed.



*Figure 10:*  
*Parent-Child Hierarchy In Analysis Services*

For the "derived” hierarchies which by definition have a fixed number of levels, you just need to create a regular "user defined” hierarchy. Thanks to the referential integrity between your different levels, you can safely define the relations between the different levels of your hierarchy in the "attribute relationship” tab. This will ensure that your user defined hierarchies will be treated as "Natural” hierarchies by Analysis Services. "Natural” hierarchies are always processed into materialized hierarchies, optimized for querying.

**Conclusion**

In this insight, we explained step by step how we can leverage the functionalities and the user interfaces delivered by Master Data Services SQL Server 2012. Master Data Service is easy to use and (only) bundled in the Business Intelligence edition. It will bring a real added value for your Business Intelligence solution by enabling the end-users to directly manage their hierarchies in a controlled way without requiring any IT intervention.

This will also be a great relief for all Business Intelligence developers who previously had to build ad-hoc solutions in order to load Excel files with master data-logic or mapping tables directly into the Data Warehouse data integration layer.

While it is easy to think of Master Data Management as a pure technological issue, only focusing on technological aspects without focusing also on the organizational aspect is likely to fail. Make sure to define governance around the ownership of your master data. The governance process can easily be implemented in Master Data Services via the use of custom security to grant access to a data steward to only a limited scope of your data and the possibility to implement workflow approval of changes using Sharepoint.