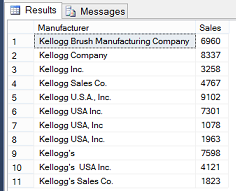
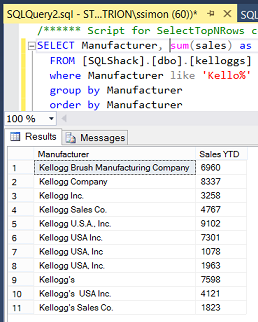
**How to use SQL Server Data Quality Services to ensure the correct aggregation of data**

**Introduction**

An interesting opportunity arose at a client site during early October which provided a phenomenal opportunity to do a Data Quality Services implementation. My client (a grocer) had been requested to produce summary reports detailing the amount of funds spent during 2016 (YTD) with the myriad of manufacturers from whom the chain purchases their inventory. All “accounts payable” entries are done manually and as such are prone to errors.



In fact, the screenshot above shows the issues encountered with a subset of “Kellogg’s”



Thus when the client would execute the above query, the total purchases for “Kellogg’s” are not truly represented. Why? Mainly because there should be one and only one summary figure for the cereal manufacturer.

This is our point of departure for today’s “fire side chat”.

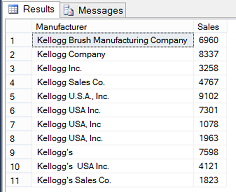
**Getting started**

For those of us whom have never worked with SQL Server Data Quality Services, please do have a look at a SQL Shack article entitled: [How clean is YOUR data!!](https://www.sqlshack.com/clean-data/)

In the above mentioned article we are shown step by step how to set up a Data Quality Services application. This includes the creation of the model PLUS step by step instructions showing us how to create and implement a Data Quality Project.

**Moving on**

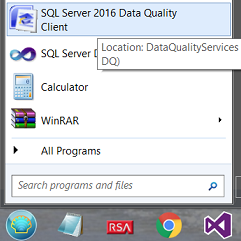
From this point onward we shall assume that we have a basic understanding of how Data Quality Services functions and the benefits that it provides. The client worked through his master list of their creditors and a portion of the list may be seen below.



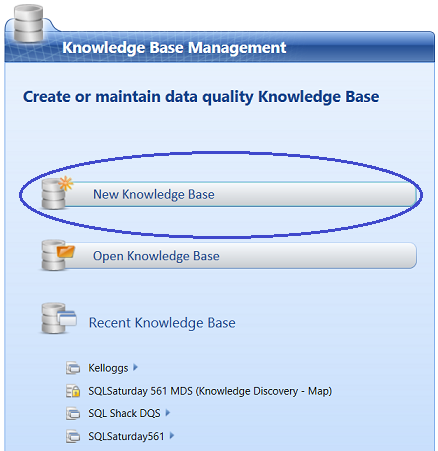
The astute reader is reminded that based upon the image above, that the purchases are not correctly aggregated as most of the variations of “Kellogg’s” could be aggregated together. However we are getting ahead of ourselves.

**Creating our Knowledge Base**

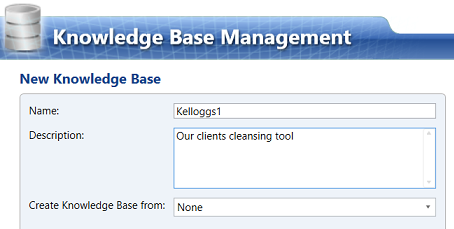
We begin by creating a new Data Quality Services Knowledge Base. Clicking upon the “Start” button, we bring up the “SQL Server 2016 Data Quality Client” (see below).



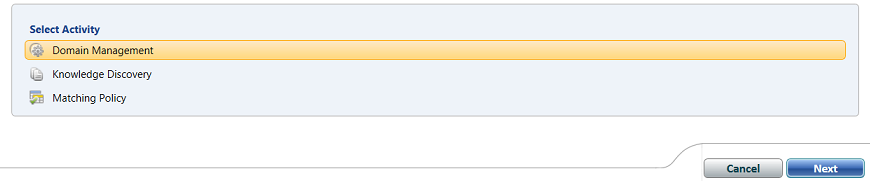
When the client appears, we select “New Knowledge Base” (see below).



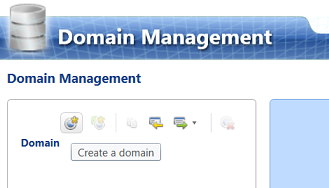
The “New Knowledge Base” designer opens.



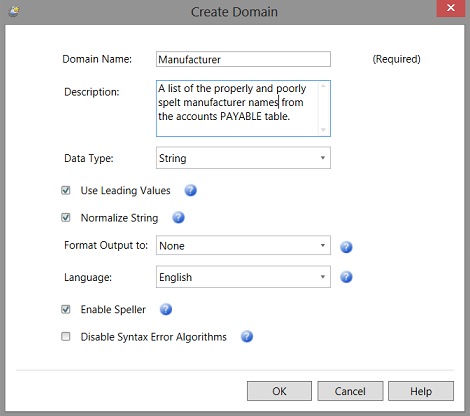
We give our new knowledge base a name and note that we do not inherit an existing knowledge base. This is why we leave the “Create Knowledge Base from:” option as “None”.

[](https://www.sqlshack.com/wp-content/uploads/2016/11/word-image-71a.png)

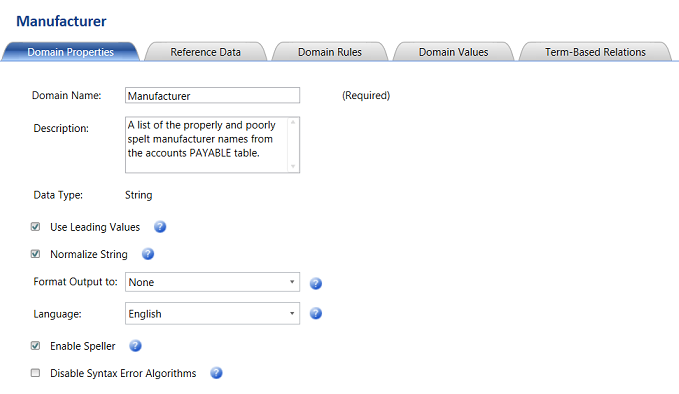
We leave the “Select Activity” set to “Domain Management” and click “Next” (see above).



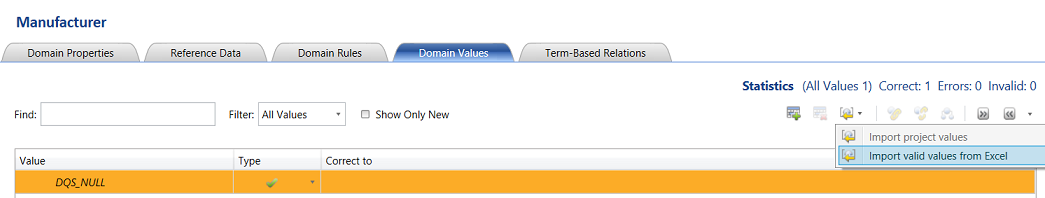
We find ourselves on the Domain Management tab. We wish to “Create a domain” (see above).



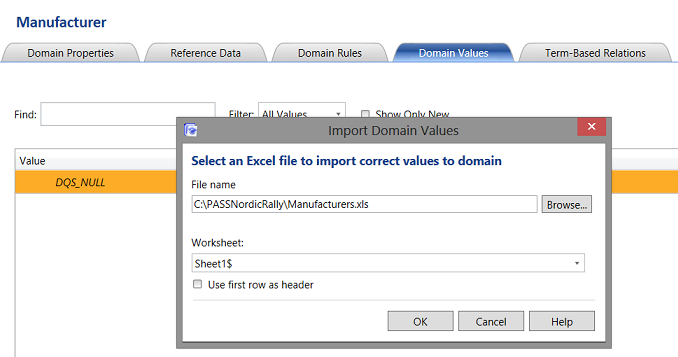
The “Create Domain” dialogue box opens. The reader is reminded that a “Domain” is comparable to a database table or an “entity” ***within*** the Data Quality Services realm. This ***domain*** will contain a list of manufacturers (a lone field / attribute). We click “OK” to continue (see above).



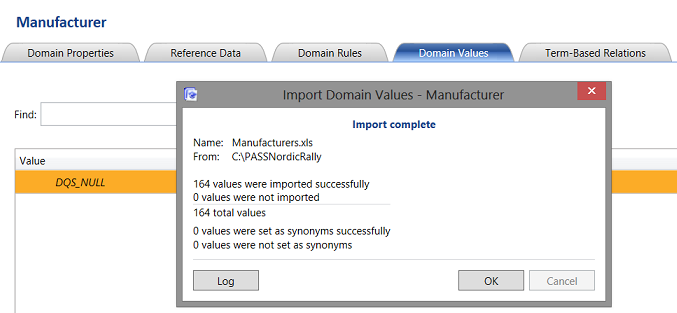
The reader will note that we find ourselves on the “Manufacturer definition screen”. There are 5 activities that may be performed upon the new domain. We shall be utilizing the “Domain Value” for this exercise. An in depth discussion with step by step instructions describing the purpose and functionality of the remaining tabs may be found in the SQL Shack article cited above.

[](https://www.sqlshack.com/wp-content/uploads/2016/11/word-image-75a.png)

Clicking upon the “Domain Values” tab we opt to import the creditor list from an Excel spreadsheet (the only way at present).



The “Import Domain Values” dialogue box opens and we browse our way to the location of the spreadsheet containing the list of manufacturers that we extracted from the clients “Account Payable” tables (see above). We click “OK” to continue.

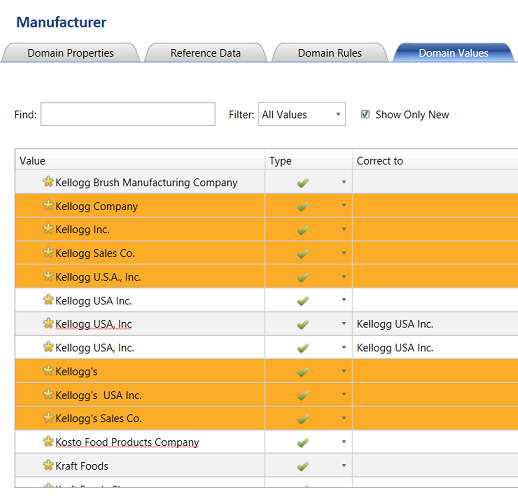


We are informed that our import was successful (see above) and click “OK” to continue.

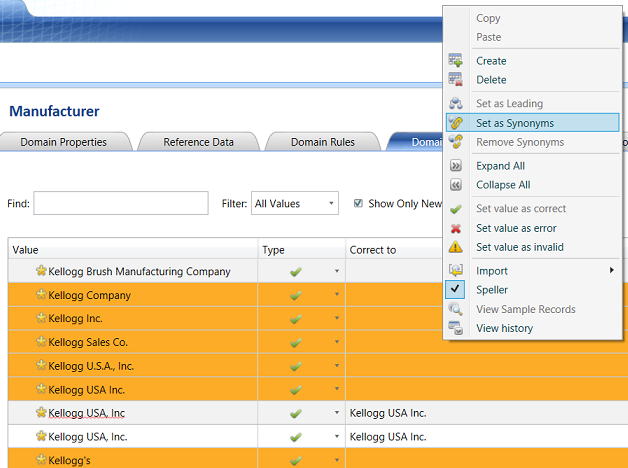


We note that all the manufacturer names now appear within the “Manufacturer” entity on the screen (see above).

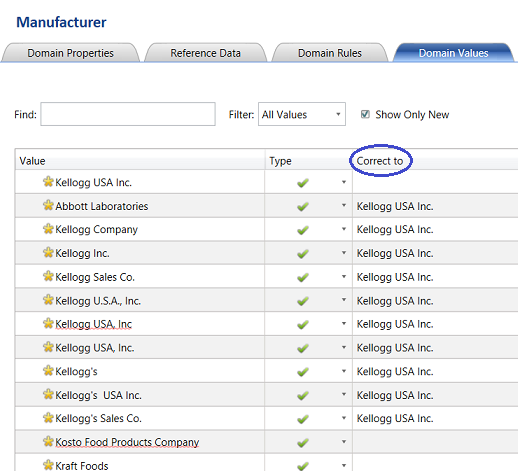
**Modifying the errors**



We highlight the wrongly spelt entries by clicking upon the row and holding down the left control key. The LAST CLICK will be on the name that we want to utilize as the correct name, which in this case is “Kellogg USA Inc.” as this was my client’s ‘name’ of choice (see above).

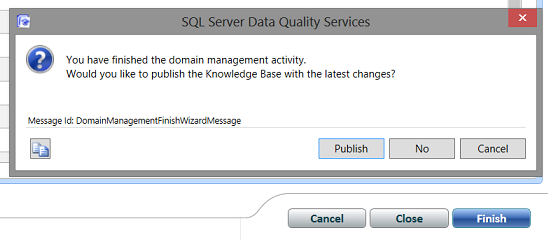


Having selected the names that we wish to alter, we “right click” on the screen and select “Set as Synonyms” from the context menu (see above).



We note that all the incorrect values now have a “Correct to” value assigned to them. This said and going forward, each time the system finds one of these “bad boys”, it will correct the name to the spelling that is shown (see above). Should you ever wish to alter this spelling, then there is no issue. Simply log into the Data Quality Services client and alter the values in a similar manner.

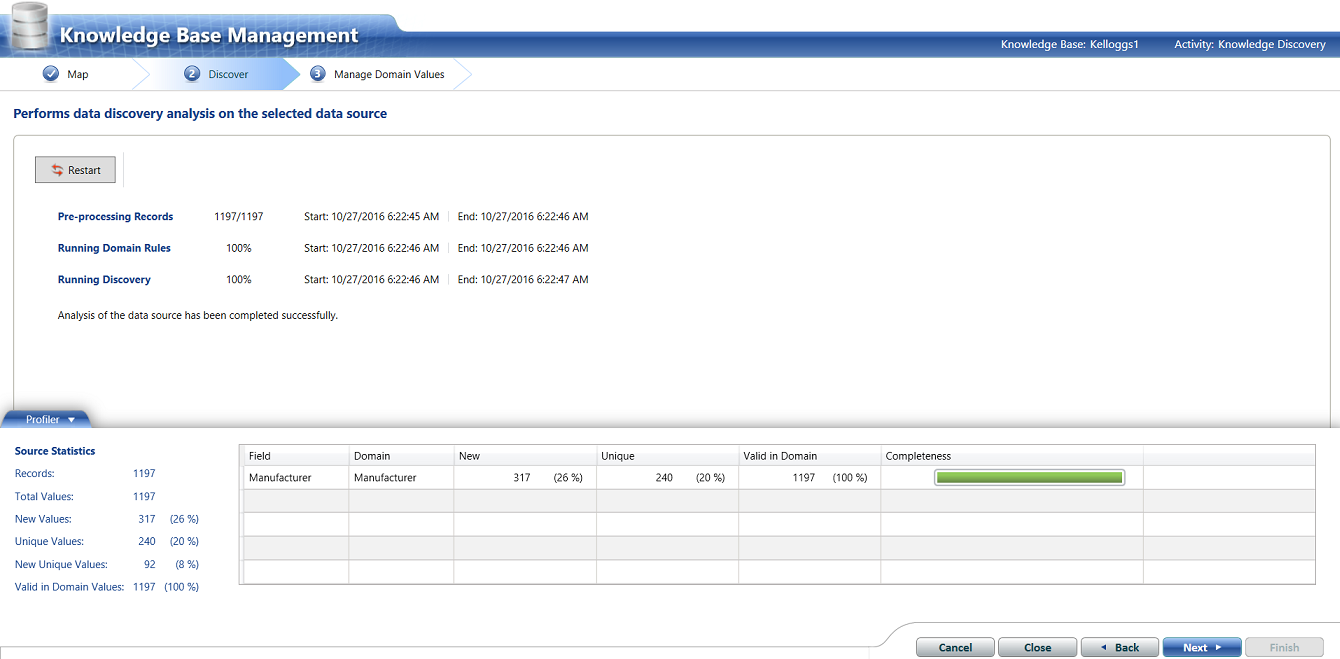
For the sake of brevity, we shall assume that this was the only error encountered. In real life, there will be numerous rows to be altered and adjusted.



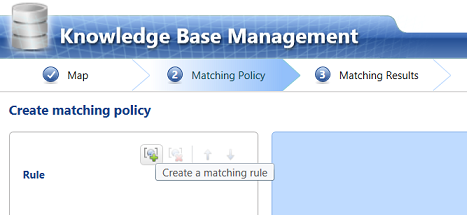
We click “Finish”. The system then asks us if we wish to “Publish” our changes. We select “Publish” and we exit the designer (see above).

Once again and for the sake of brevity our “Knowledge Base” is complete. Under normal circumstance one would run the “Knowledge Discovery” and “Matching Policy” steps as are described in detail in the SQL Shack article entitled: [How clean is YOUR data!!](https://www.sqlshack.com/clean-data/)

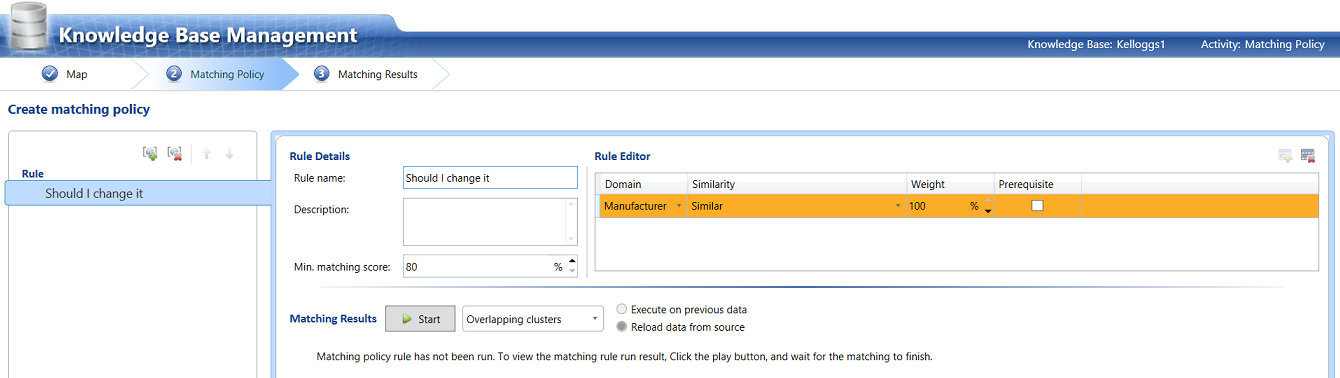
The “Knowledge Discovery” module informs you of how your selected domain values compare with an un-sampled set of manufacturers (see below).

[](https://www.sqlshack.com/wp-content/uploads/2016/11/word-image-83a.png)

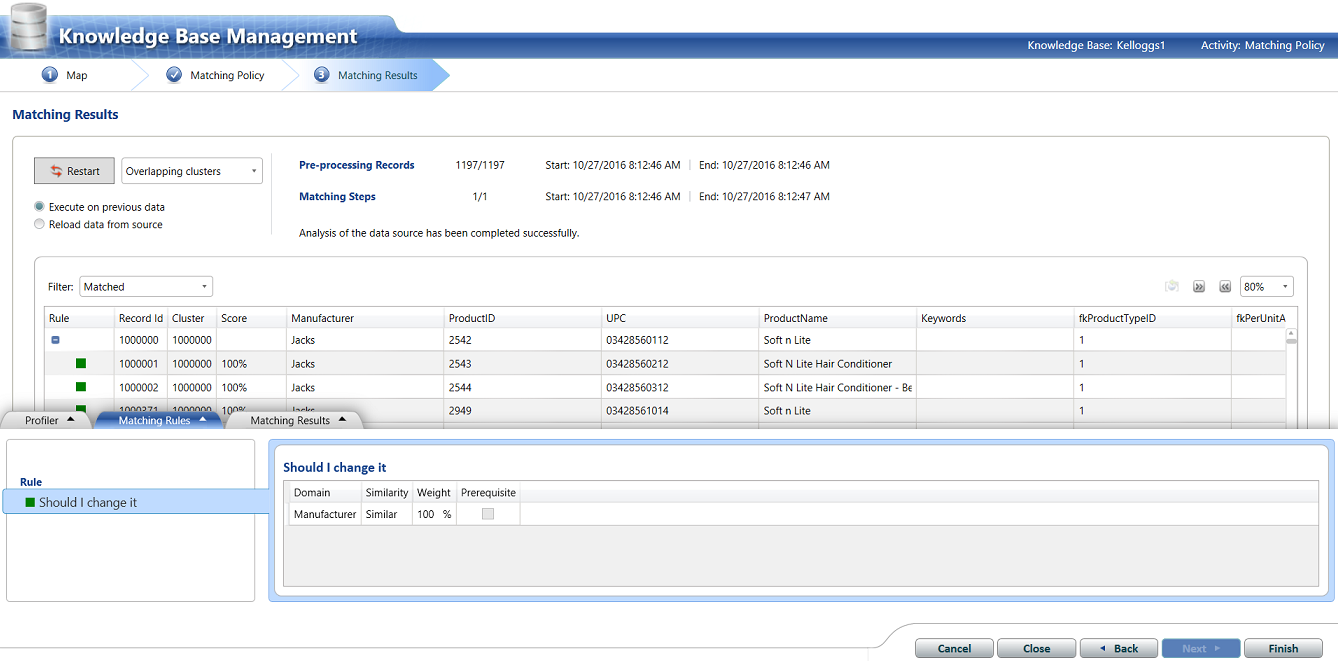
In the situation above, the astute reader will note that there are 317 **new** records. This should NOT be ***IF***the client had a complete list of creditors to begin with. Thus it is “back to the drawing board” until there are no new manufacturers (see above).



The third step is to create a “Matching Policy”. In other words just how similar must a potential wrongly spelt name be, in order to warrant changing that misspelling to “Kellogg USA Inc.” After all “Stationary” and “Stationery” are similar in spelling HOWEVER they have two entirely different meanings and therefore would never be interchanged. This unfortunately is where the “human element” comes into play, at least until the system has learnt what may be changed and what may **not** be changed.

[](https://www.sqlshack.com/wp-content/uploads/2016/11/word-image-85a.png)

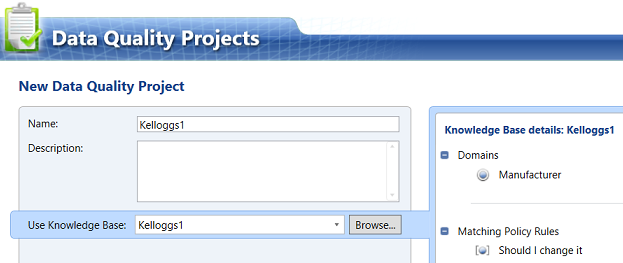
In the screen shot above we indicate to the system that we want the manufacturers to be 100% similar before effecting a change. Now this may seem like an oxymoron HOWEVER do not forget that during the first phase of this exercise we supposedly extracted ALL THE VARIATIONS in spelling plus we told the system how we wanted it to handle each specific variation of spelling. This said and assuming that there are no new variations in spelling with each successive data load, then one may assume that this rule will be efficient and effective. Obviously maintaining the “Master Knowledge Base dictionary” is of major importance.

[](https://www.sqlshack.com/wp-content/uploads/2016/11/word-image-86a.png)

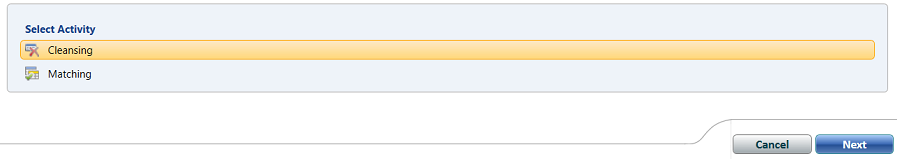
Having created our rule, we click “Next” and process the incoming data. This incoming **new** data (albeit new production data or a new subset of existing data) is done JUST TO ensure that the master list is complete. We must remember that if we encounter new manufacturers with each dataset then we do have an issue and we are therefore obliged to update the master list (as discussed above). In short and at all times, we must be certain that there are no new variations introduced.

**Creating our Data Quality Project**

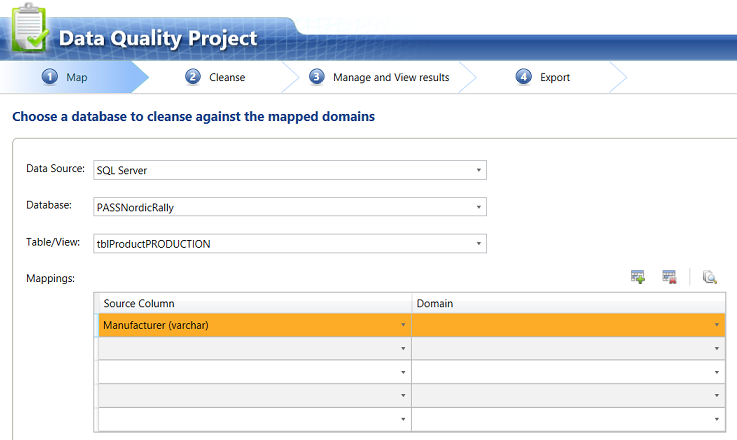
Having created and processed our Data Quality Services Knowledge Base, we are in a position to create a Data Quality Services Project based upon the Kellogg1’s Knowledge Base (see below).



Opening the new “Data Quality **Project**”, we give our project a name. Our “Kelloggs1” project will ***inherit*** the logic from the “Kelloggs1” Knowledge Base (see above).

[](https://www.sqlshack.com/wp-content/uploads/2016/11/word-image-88a.png)

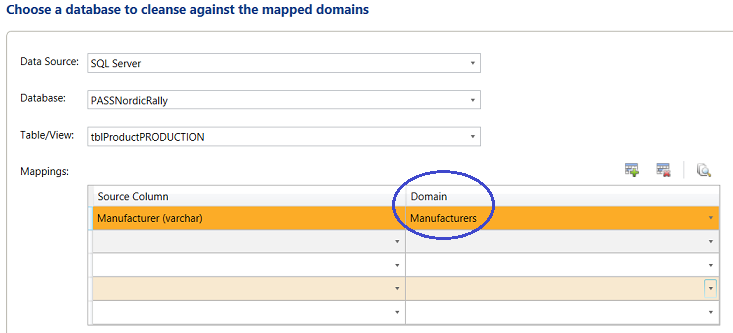
We accept “Cleansing” as the selected activity and click “Next” (see above).

[](https://www.sqlshack.com/wp-content/uploads/2016/11/word-image-89a.png)

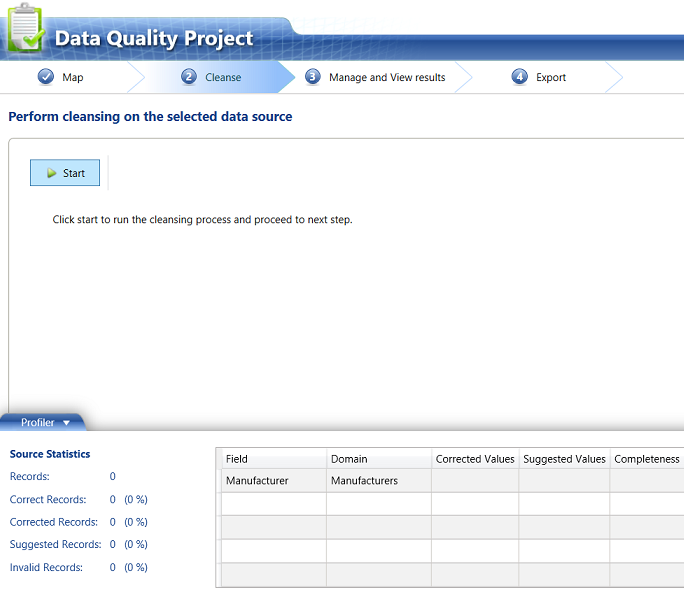
We are now asked to enter our raw data source. For this example we shall utilize a database and table that I have on my local drive. We set the “Source Column” to the manufacturer name (see above). The actual data may be seen below.



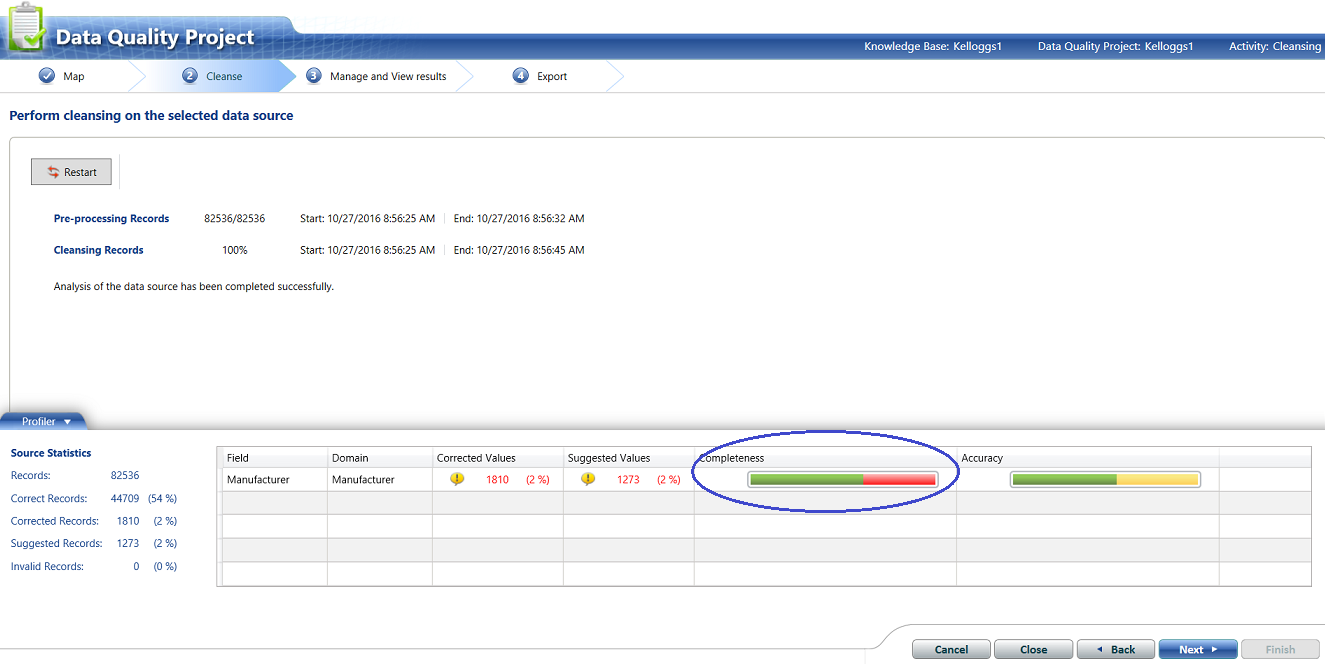
The “Domain” on the other hand, originates from the “Kelloggs1” Knowledge Base (see below).

[](https://www.sqlshack.com/wp-content/uploads/2016/11/word-image-91a.png)

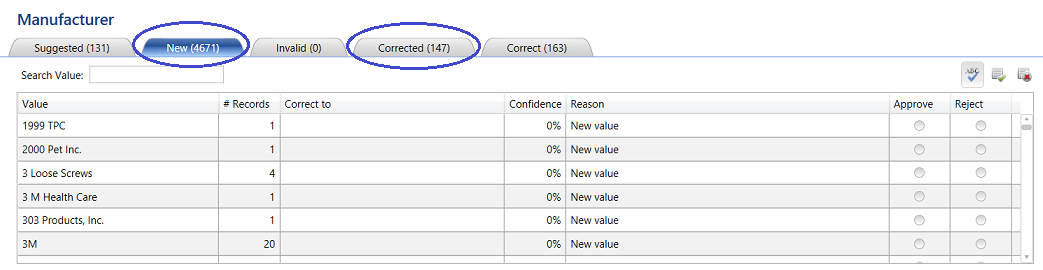
We click “Next” to continue.



At this point it is worthwhile remembering that whilst we have the Knowledge Base, each day brings new data to the enterprise. This implies more data entry and unfortunately more potential for the “gremlins” to set in. We click “Start” to process the data. Once again to check for “New” records.

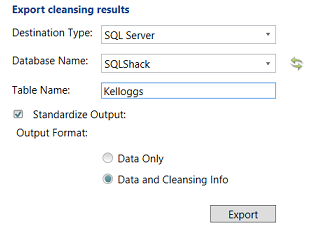
[](https://www.sqlshack.com/wp-content/uploads/2016/11/word-image-93a.png)

We note the completeness gage shows some questionable data. This is one sure indication that you are missing records within the master list. These missing “manufacturers” must be added.

[](https://www.sqlshack.com/wp-content/uploads/2016/11/word-image-94a.png)

Clicking the “Next” button we confirm this. Indications are that the new lot of incoming data contained 4671 records that were ***totally new*** and that these records had to be added to the master list. HOWEVER what is important to note is that the system found 147 records that were spelt incorrectly and **“corrected”** these records.

Having added these 4671 records to the master list and having re-published the Knowledge Base we are in a position to finish the “Kellogs1” project.



We extract the data processed via the project and export it to the “Kelloggs” table within the SQL Shack database. The importance of having this data within the database will be seen in a few minutes.



We note that the system found a plethora of records containing variations of

“Kelloggs” and in each case “corrected” the spelling (see the “Manufacturer\_Output” field above).

Now that we have completed the necessary work within Data Quality Services we are in a position to actually produce the require data to satisfy the end client’s needs.

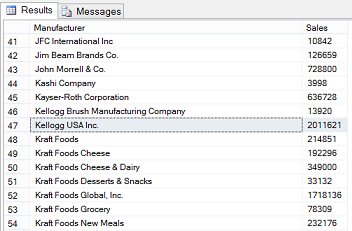
**Moulding our data to suit our needs**

As a reminder to the reader, the original list of sales values looked similar to the one shown below:



Opening SQL Server Management Studio we create a query based upon the accounts payable data and the cleansed data which originated from the Data Quality Service Knowledge Base.

|  |
| --- |
| SELECT  Kell.Manufacturer\_Output as Manufacturer, sum(sales) as Sales  FROM [SQLShack].[dbo].[DQSCompanyMasterSales2016] orig inner join [dbo].[kelloggs] kell  on kell.Manufacturer\_Source = orig.Manufacturer  Group by Kell.Manufacturer\_Output  order by Kell.Manufacturer\_Output |



Running the query, we note that the data is correctly aggregated based upon the corrected manufacturer name.

Naturally this query may be converted to a stored procedure and utilized by reporting services as we have often done is past.

**Conclusions**

Thus once again we arrive at the end of our “get together”. We have seen how powerful Data Quality Services may be when it comes to ensuring correct aggregation of data. Nowhere is this more important than when we plan to work with SQL Server OLAP cubes.

Utilizing Data Quality Services is an iterative process until you capture a correct master list PLUS the master list is continually changing. The one point that many people miss is that the Knowledge Base does in fact learn and it often compares its “corrected” values to new incoming data that really does look similar and hence passes the corrected value as a “Suggestion”. It is up to the Data Steward to continually monitor these “Suggested” values and indicate to the system, the action to take going forward.

Finally, whilst the whole process is highly iterative, one does arrive at a point where the system will monitor itself and will require less and less human intervention.