

Kimball Design Tip #24: Designing Dimensional In A Multinational Data Warehouse

By Ralph Kimball

If you are managing a multinational data warehouse you may have to face the problem of presenting the data warehouse content in a number of different languages. What parts of the warehouse need translating? Where do you store the various language versions? How do you deal with the open-endedness of having to provide more and more language versions?

There are many design issues in building a truly multinational data warehouse. I tried to cover them comprehensively in the Data Warehouse Toolkit book, so in this design tip we will focus only on how to present "language switchable" results to the end users, and we will not deal with

- international currencies,
- numeric punctuation,
- time zones,
- name and address parsing, or
- telephone number representations.

Our goal will be to switch cleanly among an open ended number of language representations, both for ad hoc querying and for viewing of standard reports. We also want to drill across a distributed multinational data warehouse that has implemented conformed dimensions.

Clearly, the bulk of our attention must focus on our dimensions. Dimensions are the repositories of almost all the text in our data warehouses. Dimensions contain the labels, the hierarchies, and the descriptors of our data. Dimensions drive the content of our user interfaces, and dimensions provide the content of the row and column headers in all of our reports.

The straightforward approach is to provide 1-to-1 translated copies of each dimension in each supported language. In other words, if we have a product dimension originally expressed in English, and we want French and German versions, we copy the English dimension row by row and column by column, preserving the keys and numeric indicators, while translating each textual attribute.

But we have to be careful. In order to preserve the user interfaces and final results of the English version, both the French and German product dimensions would have to also preserve the same

- 1-to-1 and many-to-1 relationships, and
- grouping logic, both for ad hoc reports and building aggregates.

The explicitly understood 1-to-1 and many-to-1 relationships should definitely be enforced by an entity-relationship model in the backroom staging area. That part is easy. But a subtle problem arises when doing the translations to make sure that no two distinct English attributes end up being translated into the same French or German attribute. For example, if the English words "scarlet" and "crimson" were both translated to the German word "rot", certain report totals would be different between the English and German versions. So we need an extra ETL step that verifies we have not introduced any duplicate translations from distinct English attributes.

The big advantage of this design is scalability, since we can always add a new language version

without changing table structures, and without reloading the database.

We can allow the French or German end user to drill across a far flung distributed data warehouse if we REPLICATE the translated versions of the dimensions to all the remote data marts. Remember that when we drill across distributed data marts, we execute the primary queries remotely. That is why the translated dimensions need to reside on each target database.

When a French or German end user launches a drill across query, each remote data mart must use the correct translated dimensions. This will be handled easily enough by the original French/German application that formulates the request. Notice that each remote database must support "hot swappable dimensions" that allow this dimension switching to take place from query to query as different language requests are made. This is easy in a relational environment and may be tough in an OLAP environment.

Although we have accomplished a lot with this design, including a scalable approach to implementing a distributed multi-language data warehouse, we still have some unsolved issues that are just plain hard:

- 1) we cannot easily preserve sort orders across different language versions of the same report. Certainly we cannot make the translated attributes sort in the same order as the root language. If preserving sort orders is required, then we would need a hybrid dimension carrying both the root language and the second language, so that the SQL request could force the sort to be preserved in the root language, but show the second language as unsorted row labels. This is messy and results in double-size dimensions, but probably could be made to work.
- 2) if the root language is English, we probably will find that almost every other language results in translated text that is longer than the English. Don't ask me why. But this presents problems for formatting user interfaces as well as finished reports.
- 3) finally, if our set of languages extends beyond English and the main European languages, then even the 8-bit Extended ASCII character set will not be enough. All of the participating data marts would need to support the 16-bit UNICODE character set. Remember that our design needs the translated dimensions to reside on the target machines.

The design of a distributed multi-language data warehouse is a fascinating and important problem. I'd like to hear about interesting approaches and intractable problems. Write to me.