

Design Tip #122 Call to Action for ETL Tool Providers

By Warren Thornthwaite

Let me start by stating this Design Tip is an observation on the general state of ETL tools, not any specific ETL product, some of which are better or worse than average.

As I was recently walking through the steps to create a fully functional type 2 slowly changing dimension (SCD) in one of the major ETL tools, it occurred to me that the ETL tools in general have stopped short of achieving their potential. Certainly they continue to add functionality, such as data profiling, metadata management, real-time ETL, and master data management. But for the most part, they are weak on the core functionality required to build and manage a data warehouse database. They haven't completed the job.

Let's look at the slowly changing dimension example that got me started on this topic. The product in question has a wizard that does a reasonable job of setting up the transformations needed to manage both type 1 and type 2 changes on attributes in the target dimension table. It even offers the choice of updating only the current row or all historical rows in the case of a type 1 change. What it doesn't do is manage the type 2 change tracking columns in the way we recommend. When a row changes, you need to set the effective date of the new row and the expiry date of the old row. I also like to update a current row indicator to easily limit a query to just current rows; I realize it's redundant with the value used as the expiration date on the current row (all rows with some distant future expiry date, such as 12/31/9999, are current rows by definition), but the current row indicator is commonly used for convenience and clarity. Unfortunately, this particular wizard will update either the dates or the current row indicator, but not both. So I have to add a step after the wizard to update the current rows.

Another problem with this specific wizard is that it doesn't provide any way to determine which type 2 columns changed to trigger the creation of a new row. This row reason information can be useful to audit the change tracking process; it can also be useful to answer business questions about the dynamics of the dimension. For example, if I have a row reason on a customer dimension, it's easy to answer the question "How many people moved last year?" Without the row reason, I have to join the table back on itself in a fairly complex way and compare the current row to the previous row and see if the zip code column changed. This is frustrating because I know the wizard knows which columns changed, but it's just keeping it secret.

I realize other ETL tools do a better job of automatically handling slowly changing dimensions. What's disappointing about the general state of ETL tools today is not the shortcomings of this particular SCD processing example, but the lack of effective support for most of the other 33 ETL subsystems, many of which we see in use in almost every data warehouse.

For example, junk dimensions are common dimensional modeling constructs. By combining multiple small dimensions into a single physical table, we simplify the model and remove multiple foreign key columns from the fact table. Creating and maintaining a junk dimension is not conceptually difficult, but it is tedious. You either create cross-join of the rows in each dimension table if there aren't too many possible combinations, or you add combinations to the junk dimension as they occur in the incoming fact table. An ETL tool vendor that really wants to help their ETL developer customers should provide a wizard or transformation that automatically creates and manages junk dimensions, including the mapping of junk dimension keys to the fact table.

The same issue applies to mini dimensions which are column subsets extracted out of a large dimension and put into a separate dimension, which is joined directly to the fact row. Like junk dimensions, mini dimensions are not conceptually difficult, but they are tedious to build from scratch every time. Unfortunately, I know of no ETL tool that offers mini dimension creation and maintenance as a standard component or wizard.

ETL vendors are often enthusiastic about demonstrating that their tools can address the 34 subsystems we describe and teach in our design classes. Unfortunately the vendors' responses are typically either PowerPoint slides or demo-ware implemented by an analyst at headquarters, rather than by an ETL developer working in a production environment.

I could go on down the list of subsystems, but you get the idea. Here is a link to an article Bob Becker wrote describing [the 34 subsystems](#). Take a look and see how much time you spend coding these components by hand in your ETL environment. Then ask your ETL vendor what their plans are for making your job easier.