Design Tip #97 Modeling Data as Both a Fact and Dimension Attribute

By Ralph
Kimball

In the dimensional modeling world, we try very hard to separate data into two contrasting camps: numerical measurements that we put into fact tables, and textual descriptors that we put into dimension tables as "attributes". If only life were that easy…

Remember that numerical facts usually have an implicit time series of observations, and usually participate in numerical computations such as sums and averages, or more complex functional expressions. Dimension attributes, on the other hand, are the targets of constraints, and provide the content of "row headers" (grouping columns) in a query.

While probably 98% of all data items are neatly separated into either facts or dimension attributes, there is a solid 2% that don't fit so neatly into these two categories. A classic example that we have taught for years is the price of a product. Is this an attribute of the product dimension or is this an observed fact? In our opinion, this one is an easy choice. Since the price of a product often varies over time and over location, it becomes very cumbersome to model the price as a dimension attribute. It should be a fact. But it is normal to decide this rather late in the design process.

A more ambiguous example is the limit on a coverage within an automobile insurance policy. The limit is a numerical data item, say $300,000 for collision liability. The limit may not change over the life of the policy, or it changes very infrequently. Furthermore, many queries would group or constrain on this limit data item. This sounds like a slam dunk for the limit being an attribute of the coverage dimension.

But, the limit is a numeric observation, and it can change over time, albeit slowly. One could pose some important queries summing or averaging all the limits on many policies and coverages. This sounds like a slam dunk for the limit being a numeric fact in a fact table.

Rather than agonizing over the dimension versus fact choice, simply model it BOTH ways! Include the limit in the coverage dimension so that it participates in the usual way as a target for constraints and the content for row headers, but also put the limit in the fact table so it can participate in the usual way within complex computations.

This example illustrates some important dimensional modeling themes:

- Your design goal is ease-of-use, not elegance. In the final step of preparing data for consumption by end users, we should be willing to stand on our heads to make our BI systems understandable and fast. That means 1) transferring work into the ETL back room, and 2) tolerating more storage overhead in order to simplify the final data presentation.

- In correctly designed models, there is never a meaningful difference in data content between two opposing approaches. Stop arguing that "you CAN'T do the query if you model it that way". That is almost never true. The issues that you should focus on are ease of application development, and understandability when presented through an end user interface.