KIMBALL GROUP KIMBALL UNIVERSITY

www.kimballgroup.com Number 30, November 3, 2001

Design Tip #30: Put Your Fact Tables On A Diet

By Ralph Kimball

In the middle of the 1990s, before the internet, it appeared that the data explosion might finally abate. At that time we were learning how to capture every telephone call, every item sold at a cash register, every stock transaction on Wall Street, and every policy transaction in huge insurance companies. It's true that we often didn't store a very long time series of some of these data sources in our data warehouses, but there was a feeling that maybe we had reached a kind of physical limit to the granularity of the data. Maybe we had at last encountered the true "atoms" of data.

Well, that view was obviously wrong. We now know that there is no limit to the amount of data we can collect. Every measurement can be replaced a whole series of more granular submeasurements. On the web, in the web logs we see every gesture made by a visitor BEFORE they check out and purchase a product. We have now replaced the single product purchase record with a dozen or a hundred behavior tracking records. The worst thing is that our marketing people love these behavior tracking records, and want to do all sorts of analysis on them.

Just wait until GPS data capture systems get embedded in our cars and our credit cards and our telephones. Every human being could eventually generate one or more records every second, 24 hours per day!

Although we cannot stop this avalanche of data, we have to try to control it, or we will spend too much money on disk storage. Many of our current data sizing plans are based on quick estimates. In many cases, these estimates seriously overstate our storage needs. The result may be either a decision to buy far too much storage, or to cancel our plans for analyzing available data.

In a dimensional modeling world, it is easy to see that the culprit is always the fact table. The high frequency, repeated measurements in our businesses are stored in the fact table. The fact table is surrounded by geometrically smaller dimension tables. Even a huge customer dimension table with millions of records will be much smaller than the biggest fact table.

By paying fanatic attention to the design of our fact tables, we can often slim them down significantly. Here are the guidelines:

1) Replace all natural foreign keys with the smallest integer (surrogate) keys possible.
2) As part of #1, replace all date/time stamps with integer surrogate keys.
3) Combine correlated dimensions into single dimensions where possible.
4) Group tiny low cardinality dimensions together, even if uncorrelated.
5) Take all text fields out of the fact table. Make them dimensions. Especially comment fields.
6) Replace all long integer and floating point facts with scaled integers, wherever possible.

As an example suppose we are a large telephone company processing 300 million calls per day. We could easily do a data sizing plan for tracking all these calls over a 3 year period based on the following assumptions:

Date/Time = 8 byte date-time stamp
Calling Party Phone Number = 10 byte string
Called Party Phone Number = 15 byte string (to handle international numbers)
Local Provider Entity = 10 byte string
Long Distance Provider Entity = 10 byte string
Added Value Service Provider Entity = 10 byte string

Dialing Status = 5 byte string (100 possible values)
Termination Status = 5 byte string (100 possible values)
Duration fact = 4 byte intege
Rated Charge fact = 8 byte float

Each call is an 85 byte record in this design. Storing three years of this raw data, with no indexes, would require 27.9 terabytes. Assuming a constant stream of data, we would need 776 GB of new storage per month just for the raw data!

Obviously, there is wasted fat in the above record. Let's really turn the screws on this one and see how well we can do. Using the guidelines from above, we can code the same information as follows:

Date = 2 byte tiny integer
Time of Day = 2 byte tiny integer
Calling Party Phone Number = 4 byte integer surrogate key
Called Party Phone Number = 4 byte integer surrogate key
Local Provider Business Entity = 2 byte tiny integer surrogate key
Long Distance Provider Entity = 2 byte tiny integer surrogate key
Added Value Service Provider = 2 byte tiny integer surrogate key
Status = 2 byte tiny integer surrogate key (combination of Dialing and Termination)
Duration fact = 4 byte integer
Rated Charge fact = 4 byte scaled integer

We have made a few assumptions about the data types supported by your particular database. We have assumed that the 65,536 possible 2 byte tiny integer keys are enough to support each of the dimensions where listed above.

With this design, the raw data space required by our fact table becomes 9.2 terabytes, a saving of 67%! Our monthly data growth for raw data has dropped to 256 GB.

While these numbers are still big, they give us some breathing room. Be a fanatic about designing your fact tables conservatively. Put them on a diet.