

Design Tip #81 Fact Table Surrogate Key

By Bob Becker

Meaningless integer keys, otherwise known as surrogate keys, are commonly used as primary keys for dimension tables in data warehouse designs. Our students frequently ask us - what about fact tables? Should a unique surrogate key be assigned for every row in a fact table? Although for the logical design of a fact table, the answer is no, surprisingly we find a fact table surrogate key may be helpful at the physical level. We only recommend creating surrogate keys for fact tables when certain special circumstance described in this design tip apply.

As a quick reminder, surrogate keys are meaningless (aka not-meaningful, artificial, sequence numbers, warehouse, etc.) keys, typically defined as an integer data type, and sequentially assigned by the data warehouse team to serve as the primary keys of the dimension tables. Surrogate keys provide a number of important benefits for dimensions including avoiding reliance on awkward "smart" keys made up of codes from the dimension's source systems, protecting the data warehouse from changes in the source systems, enabling integration of data from disparate source systems, support for type 2 slowly changing dimensions attributes, space savings in the fact tables when these dimension keys are embedded in the fact tables as foreign keys, and improved indexing and query performance.

But in a fact table, the primary key is almost always defined as a subset of the foreign keys supplied by the dimensions. In most environments this composite key will suffice as the primary key to the fact table. There is typically no advantage of assigning a surrogate key to the fact rows at a logical level because we have already defined what makes a fact table row unique. And, by its nature, the surrogate key would be worthless for querying.

However, there are a few circumstances when assigning a surrogate key to the rows in a fact table is beneficial:

1. Sometimes the business rules of the organization legitimately allow multiple identical rows to exist for a fact table. Normally as a designer, you try to avoid this at all costs by searching the source system for some kind of transaction time stamp to make the rows unique. But occasionally you are forced to accept this undesirable input. In these situations it will be necessary to create a surrogate key for the fact table to allow the identical rows to be loaded.
2. Certain ETL techniques for updating fact rows are only feasible if a surrogate key is assigned to the fact rows. Specifically, one technique for loading updates to fact rows is to insert the rows to be updated as new rows, then to delete the original rows as a second step as a single transaction. The advantages of this technique from an ETL perspective are improved load performance, improved recovery capability and improved audit capabilities. The surrogate key for the fact table rows is required as multiple identical primary keys will often exist for the old and new versions of the updated fact rows between the time of the insert of the updated row and the delete of the old row.
3. A similar ETL requirement is to determine exactly where a load job was suspended, either to resume loading or back put the job entirely. A sequentially assigned surrogate key makes this task straightforward.

Remember, surrogate keys for dimension tables are a great idea. Surrogate keys for fact tables are not logically required but can be very helpful in the back room ETL processing.

