

## Tom’s Ten Data Tips – June 2011

### Data Vault

Data Vault (DV) is an approach to data modeling that is specifically optimized for enterprise data warehouses (EDW’s). It was invented in the 90’s by Dan Linstedt, and published some 10 years later. Adoption has been slow, yet those who have embraced this new technology see clear benefits. It provides the benefits of Bill Inmon’s 3<sup>rd</sup> normal form data warehouse (DWH) models with the agility of Kimball style approaches.

#### 1. Data Vault Combines The Best Of Both Worlds

Ralph Kimball and Bill Inmon were the two most prominent early visionaries of data warehousing. For a long time they carried on a ferocious debate on what supposedly was “the best” DWH architecture. Inmon favored one centralized repository with data at the finest level of granularity. According to Inmon, this central repository should be modeled in 3<sup>rd</sup> normal form (hub-and-spoke architecture). Kimball favored multiple subject oriented star schema’s that should be connected through so-called conformed dimensions (bus architecture).

For purposes of *enterprise* data warehousing both Inmon and Kimball’s model have considerable drawbacks. Inmon’s model is too brittle: unless you get the initial 3<sup>rd</sup> normal form model “just right”, you will be facing subsequent changes (and maintenance) that are costly and cumbersome. This is caused primarily because time is modeled by appending a time stamp to the primary key. Kimball’s model, on the other hand, can deliver initial results more quickly but breaks down when you need to make changes to dimension tables. Also, problems will arise when conforming dimensions proves problematic because different business functions take a fundamentally different view on the same underlying facts (see also tip# 5). Thirdly, if you get the initial granularity wrong, there’s no turning back, which causes scrap and (costly) rework. DV overcomes all of these problems. It *transcends* rather than *compromises* Inmon & Kimball’s approaches.

#### 2. Data Vault Enables “True” BI Agility

Arguably one of the most valuable features of DV is that it enables *true* BI agility. It enables agility by allowing *any* DWH project to start

creating value right from the tiniest initial loads (your first Hub and Satellite) and on. You can start almost infinitesimally small. More importantly, this initial work on very small parts of the data model can always pretty much be 100% preserved and expanded on without the creation of any “technical debt” in the model, or loss of efficiency.

### **3. An EDW Needs To *Embrace Change***

Data warehouses in the past have often been unyielding systems that could not *gracefully* deal with change. In particular for an *enterprise* data warehouse, the ability to absorb changes is a key requirement. Organizations change, they divest, they merge, and all of these changes should be *anticipated* when deciding on a DWH architecture. Departments can be assumed relatively stable, but the same certainly does not hold for the corporation as a whole. That’s why DV is so often chosen for EDW’s, it is currently probably the most flexible and versatile DWH architectures available.

### **4. Use Query Templates (Or Views) To Access Your Data**

Compared to ‘ordinary’ Star Schema’s (Kimball style data warehouses), Data Vault models can be a bit daunting due to the larger number of tables, and therefore additional joins that need to be performed. However, the types of joins that need to be performed are relatively “standard” and highly repetitive (see also tip# 8). Because of self-similarity in structures, you only need to master a few ‘kinds’ of queries. The use of Point-In-Time and Bridge tables can further simplify query access. All Data Vault tables have a standard structure (see also tip# 6), and surrogate (technical) keys are used throughout the model.

The limited number of objects facilitates automation of data mart generation and access query building. It is even possible to automate query writing via relatively simple GUI’s. By transforming the access intelligence into a query building engine you assure consistency of query paths and integrity of results. DV queries require advanced SQL skills, and can therefore be viciously error prone for the uninitiated.

### **5. There Is No Single Version Of The Truth**

One of the early ‘promises’ of data warehousing was a “single version of the truth.” This is a lie! No such thing exists. The fundamental reason why a single version of the truth cannot exist is because the same *facts* can *mean* different things to different people, and often for

very good reasons. Finance, for example, may use a different definition of “customer” from marketing and operations. There is no right or wrong customer count because they might have legitimate reasons for using different definitions of “customer.” Yet all three ought to be working from the same EDW.

In Data Vault we work to establish a single version of the **facts** (in the Vault), and these same facts may well be represented differently in data marts belonging to different departments, according to *their* specific business rules. The fact that you can mine these different business rules, and their changes over time, actually provides additional valuable information.

## **6. Stick To The Standards, And You Will Gain (In Spades) From Automation**

One of the USP’s of Data Vault modeling is that because of the limited number of elements (only three kinds of tables: Hubs, Links & Satellites), there is ample opportunity to benefit from repetition. This enables you to automate a lot of the grunt work like building loading routines or retrieval queries.

There is a (very) limited number of loading templates in DV, so once you master their commonalities and differences, you can devise an engine that generates this code. That is, **if** you consistently stick to the DV standards... You may also use a tool that does this grunt work for you, but again, that means you’ll be constrained within standards.

## **7. Minimal Cost Of Redesign Allows You To “Go Fast”**

A very powerful and valuable feature of Data Vault’s inherent agility is that redesign when you got the initial data model “wrong” is simple and relatively inexpensive. What’s more important, in DV this redesign *is always guaranteed to remain “local.”* This is in stark contrast with Bill Inmon’s initial quest for a 3<sup>rd</sup> normal form DWH which causes a cascading nightmare of key revisions to **all** child tables when a parent table requires changing. Kimball DWH models have redesign problems, too, in that your initial ‘guess’ of which attributes to include in any given star schema is not conducive to redesign at all. This implies that when you’re deciding under uncertainty you’ll either include too many attributes and waste precious storage and CPU’s, or too few and face cumbersome redesign options. This threatens agility.

It seems we *always* design data warehouses under uncertainty, and in particular EDW's are prone to changes. The fact that you can mitigate redesign costs by opting for a DV has been one of the most compelling reasons for architects to go this way. The future is inherently uncertain, and requirements analysis preceding DWH design invariably carries considerable uncertainty. The last thing you want is to have these factors hamper the speed of your development efforts. Delivering value early and often is a cornerstone of Agile BI.

### **8. Data Vault Models Are Like Fractals**

Fractals are a mathematical phenomenon that is characterized by self-similarity and scale invariance. Late Benoit Mandelbrot (1924-2010) devoted a large part of his scientific career to research on fractals. Similar to fractals, a Data Vault model can grow indefinitely, without ever changing its intrinsic structure. It is the same "formula" (expanding Hubs, Links and Satellites) that keeps repeating itself.

### **9. Business Keys "Isolate" A DV From Source System Changes**

An important concept in DV modeling is the use of so-called "business keys." Business keys are the (set of) identifying attributes that are used to uniquely identify significant entities by *business people*. A business key could be one, or a set of fields combined. This could be social security number to identify taxpayers, or the compound set of street name, zip code and house number to identify a household. A VIN number to identify a motorized vehicle, or an SKU-code to identify inventory in a supermarket. These fields may or may not coincide with the primary key of tables in which these attributes reside. Sometimes the business keys play the role of surrogate keys. Discovering what *are* the business keys comes from information analysis.

The power of building a DV model squarely around business keys, is that it "isolates" the data model from future changes in source systems. If the business decides to update their ERP system, they are (very) likely to continue to use these same business keys, whereas "technical keys" in the old ERP system are highly prone to be discarded after a system change. This practice of modeling business keys thus prevents change in the warehouse as a result of changes in underlying source systems.

## **10. Data Vault Is *Not* The Holy Grail**

Although the Data Vault may well be considered *the* most important innovation in data warehousing since “the great debate” (Inmon vs. Kimball), it’s still no silver bullet. Data warehousing is *still* hard work, poorly predictable, and fraught with challenges. Many of the problems spring from the fact that the warehouse is often the first time data from disparate source systems are confronted. The DV model has an elegant way for *capturing* these data, but resolving conflicts with regards to *interpretation* remains difficult. This is largely “people work” where diplomacy, business alignment, and a vision on information provisioning and utilization are key.