



## What to do with bad data ? when & how to dispose of it

Tom Breur  
September 2009

### Introduction

Everybody hates poor quality data. But let's face it, in most organizations you *will* run into data quality issues, at least from time to time. So instead of arguing *against* poor data quality, a more useful question is "how do you *deal* with it?"

To make business intelligence (BI) applications add value to the corporation some level of data integration invariably takes place. This might be a data warehouse (DWH), operational data store (ODS), enterprise resource planning (ERP), customer relationship management (CRM) application, or what else you might have. We'll assume here that the data quality problems *originate* in upstream (primary) systems that are generating your source data. In this paper we'll focus exclusively on DWH solutions.

There are two fundamentally different ways of dealing with bad data: either you load all of the data "as is" (and deal with the errors later), or you clean/scrub the data on the way *in* to the DWH. The former is the approach as advocated by Data Vault architects, the latter I will label "Ralph Kimball's" approach, in honor of his extensive writing on this subject. In this paper I'll go over the pro's and con's of both approaches.

Two perspectives: clean on the way *in* or *out* of the DWH

We distinguish two fundamental architectural options for dealing with bad data. You either clean the data *before* they enter the DWH, or alternatively, you load the bad data and subsequently deal with the errors *on the way out* of the DWH. Note that both approaches will typically use some kind of data staging *prior* to loading data in the DWH.

The "traditional" way has been that bad data were intercepted on the way in, and that they would be dealt with during the extract, transform, and load (ETL) phase. This approach we will call the Kimball approach. In all fairness, several authors have written about this, but none so clearly and conclusively as Ralph Kimball. His book *The Data Warehouse ETL Toolkit* (2004) and a whitepaper (2007) titled "An Architecture for Data Quality" are two excellent examples.

We'll label the alternative approach a Data Vault architecture, as put forward (mainly) by Dan Linstedt (e.g.: *The Business of Data Vault Modeling* - 2008). Under this architecture you load 100% of the data, all the time. After the initial DWH load, only *new* records are added (deltas), but again, all of them. Note that an explicit choice is made to load data that is known to be of poor quality. So the DWH houses the good, the bad, and the ugly data. Only when data are moved to data marts (DM's) do you separate the "good" (which should appear in corporate reports) from the "bad" data (which are recorded in error marts). So as data are accumulating in the DWH, the *entire* history of erroneous records is preserved there.

What is a Data Vault ?

A Data Vault is a hybrid data modeling approach for enterprise data warehouses (EDW's). It combines the best of 3<sup>rd</sup> normal form and star schema modeling, specifically catered to EDW's. The benefits are greater flexibility and agility, and therefore superior resilience in the face of changing requirements. It (uniquely) provides auditability of data, which is dearly needed in this era of compliance and governance.

A Data Vault differs from traditional approaches (among others) in that known "bad" data are deliberately loaded into the DWH. This is one of the requirements for auditability. If you change (or delete) data on the way *in* to the DWH, the relation with source systems is lost, and hence auditability goes out the door. But more importantly, the philosophy behind this practice is that there is genuine value in storing "bad" data (we'll discuss an example later), and that the perspective of what constitutes "bad" data is likely to evolve over time. Therefore, you need to load all data "as is" to maintain an integral view of an authentic history of the facts. What the business considers to be "the truth" changes over time, and a history of business rules that determine what is and is not "good" data may separately contain tremendous value.

Cleansing of data, separating what data should and should not appear in corporate reports, takes place when data are moved from the EDW to DM's. Consequently, end-users do *not* query the EDW directly using ad hoc queries. They are provided access to data either through DM's or standardized (and optimized) queries on the EDW. This is how you ensure that information that shows up in reports matches management's opinion (perception) of what is considered "the truth."

Please note there is considerably more to a Data Vault, but for the purpose of this paper we'll limit ourselves here to aspects as they pertain to dealing with data quality.

## How Kimball deals with data quality

In a Data Vault architecture, the EDW contains *all* the erroneous data. This is in marked contrast with the Kimball approach to dealing with data quality problems which cleans data on the way *in* to the DWH. You flag which records are suspect/changed, and set up business rules to determine "the best" replacement value. For this data imputation you might well use advanced statistical models. A separate error dimension contains the details pertaining to erroneous and replaced records. Faulty data need to be replaced because every star in a DWH, supporting some business process, might be queried through a link to other star schemas using conformed dimensions. This DWH modeling approach works on the assumption that all the data in the DWH can be trusted, all the time.

Of course, despite your best efforts, sometimes "bad" data may still inadvertently enter a "Kimball" style DWH. If analysis later reveals there were errors left in the DWH this is seen as a shortcoming of the ETL phase. Hopefully feedback about errors will help to resolve these problems in the future. The DWH set-up, from both a technical as well as a process standpoint is to *only* add data, and basically *never* remove or overwrite data in a Kimball DWH. If you *would* overwrite existing data this will cause inconsistencies in corporate reports, some of which may already have been (widely) distributed. Altering data after it has been committed to the DWH is indeed a very rare occurrence. It is inherently "impossible" to change your mind about which data are "good" or "bad", and are presented as such to the end user community.

## Why store "bad" data ?

The Data Vault practice of deliberately loading "bad" data sometimes feels counter intuitive. Yet the reasons are very compelling. To begin with, "bad" data is rarely a purely technical problem. In most cases, bad data are a sign of something, somewhere going wrong. This is what Michael Hammer referred to in *Reengineering the Corporation* (1994): "seemingly small data quality issues are, in reality, important indications of broken business processes." Maybe an interface doesn't provide employees some necessary option and therefore they resort to "rigging" the system. We have all stood in line, listening to a clerk saying: "hang on, the system doesn't allow me to do this." Front-office staff *will* make it happen, if circumventing the system in some way is required. Even if this means manipulating data to "force" a transaction to be completed. The relation between data quality errors and broken business processes can come in myriad forms and shapes.

The second reason for deliberately storing "bad" data is more subtle. When you overwrite (or delete) erroneous records on the way *in* to the DWH, you need to make that call then and there (in a Kimball DWH). When you store "bad" data "as is", you are in a position to reconsider

what constitutes “good” or “bad” data, and these definitions can (and often will) evolve over time.

One company we worked with set their DWH processes so that a central CRM system would be the first and sole place where customer ID’s were to be assigned. Since all systems and business process were geared that way, this worked remarkably well, and led to very high standards of data quality. However, a very small percentage of records got “lost” every month because they had no customer ID’s, so these were dismissed. Further analysis revealed these were exception prospects that entered through a so far unacknowledged sales channel.

When these records without a customer ID were dropped, the company literally “dropped the ball.” These customers were effectively ignored, as if they did not even exist. When the source of these errors was found, they managed to capitalize on this business opportunity, and grew it into a considerable revenue stream. This, is a perfect example of how supposedly “bad” data holds value. It can be like a diamond in the rough, waiting to be discovered and exploited. They first discovered a broken business process (prospects that were not getting follow-ups), and fixed that. Later this proved to be a viable new business venture, that was created out of “nowhere”: prospects that supposedly were not allowed to exist.

## Conclusion

“Bad” data will be with us to stay for *at least* the foreseeable future. Because errors often only become apparent when you try to *integrate* data, a DWH is particularly susceptible to run into those kinds of challenges. Installing robust processes to deal with this reality are therefore a necessity, a key to success with your DWH.

We have distinguished two fundamental approaches to dealing with poor quality data. Either you delete or overwrite bad data on the way *in* to the DWH (the “Ralph Kimball” style), or, you load bad data “as is” and apply transformations (and filters) when you move data to DM’s (“Data Vault” style). Both approaches have pro’s and con’s.

An advantage of dealing with data quality errors on the way *in* to the DWH is that your architecture requires one less layer of storage, and that EDW data are available to see (and presumably be trusted) by everyone in the corporation. The disadvantage is that you are stuck with any choice you make with regards to how you treat erroneous records. And you cannot mine how business rules that transform “bad” data into “good” data evolve over time.

The advantage of the Data Vault architecture is that all value inherent in storing bad data remains available as a source of learning and

improvement. You retain maximum flexibility in how you choose to treat “bad” data. As your insights evolve and mature you can constantly update your transformation (and selection) rules from DWH to DM. Additionally, your DWH is *and remains* forever fully auditable, and can therefore become the authoritative single source of the facts.

#### References

Dan Linstedt, Kent Graziano & Hans Hultgren (2008) The Business of Data Vault Modeling. ISBN# 9781435719149

Ralph Kimball & Joe Caserta (2004) The Data Warehouse ETL Toolkit. ISBN# 0764578578

Ralph Kimball (2007) An Architecture for Data Quality. Whitepaper Kimball Group.

Michael Hammer (1994) Reengineering the Corporation. ISBN# 1857880560