The Seven Tenets of Scalable Data Unification

by Michael Stonebraker, Chief Technology Officer, Tamr Inc.

Executive Summary

This paper defines the concept and process of data unification and compares different technical approaches to achieving the desired end-state of clean, accurate, consolidated data sets. It then proposes seven tenets that must be considered by data management practitioners who seek to unify large volumes and varieties of data.

I Introduction

Data unification is the process of:

- **Ingesting** data, typically from operational data systems in the enterprise.
- Performing **data cleaning**, **e.g., -99** is often a code for "null," and/or some data sources might have obsolete addresses for customers.
- Performing **transformations**, **e.g.**, euros to dollars or airport code to city_name.
- Performing **schema integration**, e.g., "salary" in one systems is "wages" in another.
- Performing **deduplication (entity consolidation)**, e.g., I am "Mike Stonebraker" in one data source and "M. R. Stonebraker" in another.
- Performing **classification** or other complex analytics, e.g., classifying spend transactions to discover where an enterprise is spending money. This requires data unification for spend data, followed by a complex analysis on the result.
- **Exporting** unified data to one or more downstream systems.

There are many kinds of products that purport to solve one or more of the above issues. Two popular ones are Extract, Transform, and Load (ETL) systems and Master Data Management (MDM) systems. We discuss each in turn in the next section.

1.1 Traditional Solutions

The traditional wisdom is to use an Extract, Transform, and Load (ETL) system for data unification. ETL systems have been widely used for about 20 years to load data warehouses -- typically with customer-facing data -- and have the following operational structure:

Step 1: Construct a global schema up front.
Step 2: For each data source to be integrated:

- Have a programmer interview the business owner to discover the local schema used.
- Have the programmer write the conversion routines from the local schema to the global schema.
- Have the programmer write cleaning and transformation routines.
- Have the programmer write a script to load the global schema, and then update it over time.

Given their intensive requirement for human effort, most data warehouses unify only a small number of data

sources – typically 20 or less.

Another class of products, often from the same vendors, is Master Data Management (MDM) systems. The MDM methodology suggests constructing a master record for the various entities used in the enterprise (customers, departments, parts, suppliers, etc.). A master record indicates what fields are defined for the the entity. Then, MDM products construct a repository to hold master records; in other words, they become the definitive standard for enterprise data. Since local data records in individual operational systems may identify entities in different ways, MDM system have a "merge-match" component. The idea is to merge all the individual records and then match the various records that belong to the same entity. The popular MDM systems suggest using rules to perform this matching. Hence, an MDM user would write a collection of rules for his environment, for example:

"Dick" matches "Richard" in the name field
-99 matches null in the salary field
If systems A and B have different values for address, then use the one from system A

In effect this "fuzzy merge" will produce a collection of "golden records," one for each entity.
Think of an MDM system as establishing ground truth in advance and doing a particular form of ETL.

1.2 Scope of this Paper

In this white paper we discuss one aspect of data unification systems that is often very important, namely scalability. On the one hand, if your problem is to unify three data sources, each of which has ten records, then it doesn't much matter what kind of a tool you use. In fact, a whiteboard or paper and pencil may be the best technical approach. On the other hand, there are many larger unification problems. In the next section we discuss two example problems from Tamr customers.

1.3 Example Problems

The first example comes from Novartis, a large drug company. They have a need to unify customer-facing data (who bought what when) and use a traditional ETL system to unify this data into a data warehouse. They are quite happy with this solution, which unifies a small number of data sources. However, they are "breaking their pick" on a second unification problem, namely the unification of experimental data.

Novartis has around 10,000 bench scientists in several countries doing "wet" biology and chemistry and writing their results in "electronic lab notebooks." Novartis wants to connect scientists who are using the same reagents to produce different results or who are producing the same result using different reagents or different techniques. In other words, they wish to enhance social networking among their professionals. Effectively, they wish to unify 10,000 lab notebooks. Unfortunately, these data sets are in different languages and do not benefit from any standard ontologies or measurement units. In effect, Novartis has a data unification problem at scale 10,000. This is three orders of magnitude larger than the data sources typically unified by an ETL product.
The second example concerns General Electric (GE) procurement systems. A procurement system is used whenever an employee wants to purchase something, such as a paper clip. It will print the employee a purchase order that (s)he can take to the neighborhood Staples to obtain the desired goods. An ideal enterprise has a single procurement system; however, in large enterprises a single system is rarely the case. For example,

GE currently has about 80 procurement systems, each of which has an independently constructed supplier database. The average number of suppliers in a GE procurement system is about 17,000. One might wonder why there are so many systems, and the answer is quite simple. GE has many independent divisions; often these have resulted from acquisitions. An acquisition, of course, comes with a procurement system. Any enterprise could require all of its divisions to use a single system, but this would slow down progress in each division dramatically. Hence, many large enterprises allow business units to be relatively independent, which leads to the multiple-system phenomenon noted above.

There is a huge upside to GE to performing data unification on these 80 supplier databases. When the Staples contract comes up for renewal, a procurement officer would love to know the terms and conditions negotiated with Staples by other business unit, so that (s)he can demand "most favored nation" status. GE estimates that accomplishing this task would save GE around $1B per year. To get this benefit, 80 data sets with a total of 1.3M+ records must be unified. This is five orders of magnitude larger than the paper and pencil case noted earlier. Scaling to these kind of numbers obviously cannot be done with paper and pencil or a whiteboard.
In this white paper, we discuss what is required to solve these large unification problems. The following section discusses seven tenets that any scalable system must conform to. As far as we know, Tamr is the only data unification system that conforms to all of the tenets.

II The Seven Tenets of Scalable Data Unification

2.1 Automatic Operation

Any unification operation, when performed at scale, must have its principal component be automatic operation, with a human handling exceptional conditions. There can be NO primarily manual operations.

This obvious statement can be demonstrated by a back-of-the-envelope calculation. Suppose, for example, we want to deduplicate the 1.3M+ GE records noted above. Even at 1 record per minute, this will take an analyst more than 10 years. To get this done in a week will take a team of some 500 people. Training a team this size to make consistent choices will take months of planning, not to mention locating a team of this size. Obviously, this is not likely to happen.

Hence, any scalable unification system must be automatic, and expect to have a human review only a small fraction of the automatically performed operations. Anything else will collapse at scale. The techniques to perform automatic integration are further discussed in some of the following tenets.

> Tenet 1: Any scalable system must perform the vast majority of its operations automatically.

2.2 Schema First

Imagine the problem of unifying the Novartis data. Traditional ETL systems require the target schema upfront. Hence, we call this kind of system "schema first." However, there is no possible way any human can locate and understand 10,000 data sources. Imagine that a human could understand a data source every 15 minutes. At this rate, a human would require 18 months to perform this task. As a result, no Novartis employee knows what

the global schema looks like.

I am reminded of the late 1990s, when a popular enterprise project was to create an "enterprise-wide global data model." A team of several people would be dispatched to work on this project and might return with an answer in a couple of years. By then, the enterprise business situation will have assuredly changed, and the model is obsolete. To the best of my knowledge, all such projects failed, and have rarely been attempted since then. The only feasible option is to build a global schema bottom-up from the local data sources. This amounts to "schema last." Also, Tenet 1 must also be adhered to; at scale, schema integration must be primarily automatic. Any manual system (such as a drag-and-drop user interface that indicates that an attribute in one data source is the same as another attribute in a second data source) clearly won't scale.

> Tenet 2: Schema-first products will never scale. The only option is to run a schema-last product.

2.3 Collaborative System

Next consider the "write cleaning routines" of the ETL algorithm in Section 1. Consider an actual Novartis situation. They have a particular representation for certain genomic data, which they call an EC50. Also, consider an EL50. This can either be:

An EC50, and there is an error in naming, or
A new kind of genomic representation

A programmer running the ETL framework has no clue which of the above cases is true. There is only one person who can decide what an EL50 is: a genomic domain expert. There are thousands of expert scientists at Novartis, who can decide this question; however, none work for the IT organization, which is the organization who would construct and run an ETL framework.

In summary, any data unification system must depend on two kinds of people:

- Professional computer scientists, who can set up and run a data unification pipelined
- Domain experts, who can resolve data cleaning (and other) issues, that require human involvement

Any successful data unification project must depend on both kinds of people, which we term a **collaborative system**. Any data unification system with only a single kind of user has no chance of scaling when domain-specific cleaning must be done.

> Tenet 3: Only collaborative systems can scale when domain specific operations are required.

2.4 No N ** 2 Operations

Consider now the GE supplier deduplication problem, which has 1.3+ M local suppliers. A system must decide how many entities the following three supplier records correspond to:

Name	Address	Type of Business	#Employees
IBM Corp	Armonk N.Y.	Computers	100,000
IBM Corp	Armonk N.Y.	Data Systems	90,000
International Business Machines	New York, N.Y.	Computers	95k

Of course, a human can look at these three records and decide they correspond to a single entity. However, a data unification system must do so automatically. The standard wisdom is to cluster all records in a multi-dimensional space formed by the four attributes, with a heuristically specified distance function. Records that are close together in four space are probably the same entity. There are many clustering algorithms, but all have complexity N ** 2, where N is the number of records.

Clustering 1.3M+ records in this fashion is an O(2 * 10 ** 12) computation. A moment's reflection indicates that if we can compare two records in 20 microsecond, then this computation will take around two years on a typical personal computer. Two observations are readily apparent from this example.

> Tenet 4: To scale, any unification computation must be run on multiple cores and multiple processors.

> Tenet 5: Even with Tenet 4, a parallel algorithms with lower complexity than N ** 2 is required for truly scalable applications.

2.5 Not Based on Rules

We now explore another GE example, which illustrates the problems with rule systems. GE wants to classify "spend transaction" and associate each with a node in a pre-specified classification hierarchy. For example, the root node might be "parts," and a descendent node could be "computers," while a third-level node could be "memories."

GE wishes to classify 20M such transactions, and begins by writing classification rules, with some assistance from Tamr. One such rule might be:

Any transaction from IBM is classified as a computer transaction.

With 500 rules, Tamr and GE managed to classify 2M transactions. Note that this is about 10% of their problem and 500 rules is about the upper bound of what a human can understand. Obviously, to classify the 20M transactions in this fashion, one would require at least 5000 rules. I have never seen a rule application this

large. Put differently, rule systems simply don't scale. Instead, the actual purpose of the rules was to generate a collection of matches between a record and a specific classification. These matches were then used as training data by the Tamr machine learning system which classified the remaining 18M transactions.

> Tenet 6: A rule system implementation will not scale. Only machine learning systems can scale to the level required by large enterprises

2.7 Real-time and Streaming Operation

Consider the GE procurement example once more. They have tried, without success, on multiple occasions to retire their various procurement systems and replace them with a single system. As a result, their current Tamr project is not replacing procurement systems; instead it sits off to one side and is queried by procurement officers in order to support better decision making. In effect, it is an analytical system to support better decision making.

Therefore, whenever any procurement system is changed, the inserts, deletes or updates must be forwarded to Tamr. Hence, GE has a "batch" problem to unify their supplier databases at the initial time. Then, they have an "incremental" problem to update the unified result as supplier records are added, deleted or modified in the source systems.

In general, source systems can be updated, and they produce a sequence of "delta" records. A unification system must be capable of streaming operation, whereby delta input records stream through the system, and cause the unified output to be updated in real time. Although GE can tolerate some delay in this update process, other applications may have tighter latency bounds.

One option is, of course, to run the entire unification workflow on all of the data each time a change occurs. This "brute force" approach runs everything from time zero for every change. If changes happen once a month this strategy might make sense. However, if changes occur more frequently and the unification problem is large (i.e., unification at scale), then this brute force strategy will clearly fail, and an incremental approach is required. In other words, a unification system must be prepared to accept delta records and perform incremental unification.

> Tenet 7: Incremental unification in real time must be supported

III Summary

In this paper we have indicated seven tenets that must be present for any enterprise data unification system to scale. When considering any solution where scalability is needed, make sure that it satisfies these criteria. Generally speaking, the following conclusions are evident:

- ETL systems will fail at least Tenets 1, 2 and 3.
- Master data management (MDM) systems will fail at least Tenets 1, 2 and 6
- Self service data preparation (SSDP) systems will fail at least Tenets 1 and 3
- SIn addition, all above systems are likely to fail Tenets 5 and 7.
- The only system we know of that satisfies all seven criteria is Tamr.

About the Author

Tamr Co-founder and CTO Dr. Michael Stonebraker has been a pioneer of database research and technology for more than a quarter of a century. He was the main architect of the Ingres relational DBMS, and the object-relational DBMS PostgreSQL. These prototypes were developed at the University of California at Berkeley where Stonebraker was a Professor of Computer Science for twenty five years. More recently at MIT, he was a co-architect of the Aurora Borealis stream processing engine (commercialized as StreamBase), the C-Store column-oriented DBMS (commercialized as Vertica), and the H-Store transaction processing engine (commercialized as VoltDB). Currently, he is working on science-oriented DBMSs and search engines for accessing the deep web. He is the co-founder of six venture capital-backed start-ups.