ABSTRACT

This paper presents conceptual view of Data Vault Modeling (DVM), its architecture and future opportunities. It also covers concepts and definitions of data warehousing, data model by Kimball and Inmon and its evolution towards DVM. This approach is illustrated using a realistic example of the Library Management System.

KEYWORDS: Conceptual Data Vault, Master Data Management, Extraction Transformation Loading

INTRODUCTION

This paper explains the scope, introduces basic terminologies, explains the requirements for designing a Data Vault (DV) type data warehouse, and explains its association. The design of DV is based on Enterprise Data Warehouse (EDW)/ Data Mart (DM). A Data Warehouse is defined as: “a subject oriented, non-volatile, integrated, time variant collection of data in support management’s decisions” by Inmon (1992). A data warehouse can also be used to support the everlasting system of records and compliance and improved data governance (Jarke M, Lenzerini M, Vassiliou Y, Vassiliadis P, 2003). Data Marts are smaller Data Warehouses that constitute only a subset of the Enterprise Data Warehouse. An important feature of Data Marts is they provide a platform for Online Analytical Processing Analysis (OLAP) (Čamilović D., Bečejski-Vujaklija D., Gospić N, 2009). Therefore, OLAP is a natural extension of a data warehouse (Krneta D., Radosav D., Radulovic B, 2008). Data Marts are a part of data storage, which contains summarized data.

The ETL/ data integration process involves extraction of data from different transactional systems/sources, cleaning of the data (Involves data type match, LTRIM and RTRIM trailing spaces, eliminating bad data called the junk data), transforming the data into the appropriate
format (according to user requirements) and loading the resultant data into the preferred target
data warehouse (Larson B, 2009). During the ETL process, data from a variety of data sources
is extracted by extraction routines and then loaded into the staging area where the data is
cleansed and transformed before loading it into the data warehouse (Vassiliadis P., Simitis A.,
Skiadopoulos S, 2002; Vassiliadis P., Simitis A, Baikousi E, 2009). Meta data defined as the
“Data about the data” also plays a very important role (Adelman S., Moss L., Abai M, 2005).
Before a data warehouse can be accessed efficiently, Meta data plays a very important role, in
understanding the data availability and where it is located (Jarke M., Lenzerini M., Vassiliou Y,
Vassiliadis, 2009). Designing a Data warehouse is not an easy task. There are several hurdles a
designer has to handle for example, he or she has have to design a semantic reconciliation of
information about data from the source and produce an enterprise model for the data
warehouse (Jarke M., Lenzerini M., Vassiliou Y, Vassiliadis, 2009).

A data warehouse has two distinguished types of architecture (Inmon, 2000, 2008, 1992):

1. Bill Inmon proposed CIF (Corporate Information Factory) as an integrated DW, the
database is in the third normal form (3NF), from where the multi-dimensional data marts can be
derived.

Kimball R, 2008) proposed the bus architecture which uses ER diagram, where DW is an
assembly of data marts with conformed dimensions.

Both of the above models do not handle the changes in the system that feed to DW. Also, both
models do not function well in an unstructured environment. The traditional DW models are not
flexible to scale as the data becomes unstructured/semi-structured. To address these above
mentioned modeling requirements and issues, new modeling approaches have been
introduced, among which are DVM and Anchor Modeling (Rönnbäck L, Hultgren H, 2013)

LITERATURE REVIEW

Data Vault Modeling
Data Warehouse 2.0 is a second generation definition of standard Data Warehouse
architecture. The most important advantage in data warehouse 2.0 is its capability to support
changes in historical data (Mundy J, Thornthwaite W, Kimball R, 2008). Data modeling
techniques for data warehouses vary for operational systems and for data marts due to the
unique set of requirements, constraints and variables related to modern data warehouses.
Some of these include the needs of Kimball’s data warehouse definition: integrated, non-
volatile, time variant, subject oriented, auditable, agile and complete store of data.

The DVM technique was introduced by Dan Linstedt (Linstedt, 2000, 2008, 2011) to solve
problems of performance and flexibility, which enables maintenance of a permanent system of
data records (Jovanović V, Bojičić I., 2012). DVM is enabled by using concepts of Hub, Link and
Satellite entities. Briefly explaining these concepts, Hub entities hold identifiers (business keys),
Link entities serve as foreign key references, and Satellite entities show contextual information
that attributes to a Hub or Link. We can see satellites as a rate of change, in terms of the type of
data and source system (Hultgreen H, 2012).

Anchor Modeling is one of the techniques built on the evidence that the environment
surrounding a DW is in a constant state of change on the inside of model (Rönnbäck L, 2011,
O., Rönnbäck, et al., 2010). The goal using this modeling approach is to achieve “a highly
decomposed implementation that can efficiently handle growth of the data warehouse without having to redo any previous work”, reproducing this idea of sheltered semantic temporal data put forward in data warehouse 2.0 architecture (Regardt, et al., 2010).

The new age data models, DVM and Anchor Modeling are categorized by strong normalized data, in consideration to the changes in the business environment, adaptation to frequent changes in business, as well as the modeling of smaller parts of the data model, without needing redesign. Christopher Date used the Sixth Normal Form (6NF) in the relational database theory, to describe a database which decomposes variables to complex elements. He also points out many shortcomings while attempting to model temporalized data using the Fifth Normal Form (5NF) and then introduces the Sixth Normal Form (6NF): "A relevant (table) R is in 6NF if and only if R satisfies no nontrivial join dependencies at all, in which case R is said to be irreducible" (Knowles C, 2012; Date et al., 2002). DVMs in complex situations, consist of a Satellite table, consist of one of the attributes, and this becomes a 6NF design. A DV is more sensible in case of a disparate data source (Linstedt, 2011; Graziano, 2011). To enable tracking of data, DV does not transform data from the different sources before loading them into the warehouse; this enables a permanent system of records. But the CIF modeling techniques consolidate data well in advance.

The flow diagram (Jovanović et al., 2012), explains the evolution of DW data into DV and the derived data in the data marts. Data Warehouse state history and reporting from DW methodologies of data warehouse design are not fully formalized.

There are four classes of approaches towards Data warehouse design that can be recognized:

a. “Data driven” is based on analysis of data, database schema and available data (Inmon, 2000; Winter, R., Strauch, B, 2003; Jensen et al., 2004; Guo et al., 2006; Golfarelli M, 2008).

b. “Demand driven” starts with requirement gathering from the users, this does not include mapping of available data, which remains a problem (Kimball, 1996; Jensen, et al., 2004; Guo et al., 2006; Golfarelli M, 2008).

c. “Goal driven” (Boehnlein, M., Ulbrich v.E, 2000; Westerman, 2001; Guo et al., 2006) is based on business strategy which begins with analysis of KPI’s (Key Performance Indicator). This is a subset of demand driven.

d. Hybrid approach combines requirements from users with analysis of management and operational data. The data driven and demand driven are applied self-sufficiently and sequential to each other, while both are applied in parallel (Romero O., Abello A, 2009).

DV is a baseline method to data warehouse design and they are derived from developers’ views. One of the obstacles while building a data warehouse using different methodological approaches is the process of identifying and generating measure and dimensions (Hub, Satellite and Link in terms of DV) in an automated way (Romero O., Abello A, 2013).

**Architectural Definitions**

The DV approach has few common architectural components, which are explained in detail in the sections below. Understanding these components are the basic foundation for understanding DVMs.
DVM methodologies include each of the above stated components. Architectural components include the staging area and DV, which are explained below:

**Staging Area.** The staging area contains all the source system tables in the database to house incoming data 1:1 mapping (with some additional source system driven elements). The staging area is purged (refreshed) before each batch load cycle. In other terms, the staging area should never house the history of loads. This is commonly called “transient staging area”. Staging areas do not house referential integrity, foreign keys or the original primary key definition. They house sequential numbers, which are reset at every load and cycled for each table with each batch cycle. Staging areas also house a load date stamp and record source for each table.

There are exceptions to the staging tables loading a de-normalized COBOL based file and executing normalization (form 3 – splitting a single table into multiple tables), as these staging tables will present parent ID references. Loading a de-normalized XML based file and executing normalization form 3 will present the Parent ID reference.

The staging area can be partitioned in any format/manner. This format is generally decided on by the data warehousing team. The staged tables may contain indexes (post load), in order to improve DV loads with performance downstream. Staging area data should be backed up regularly at regular intervals or at scheduled intervals.

With DV being 100% real-time feeds, there appears to be no need for a staging area. There are already a few instances where operational DV by passes staging area in real-time. The reasons why staging areas continue to exist are:

- a. Data synchronization : also contains other static lookup data
- b. Hot-data backup : backup of data, when the queuing engine dies (transactional feed engines)
- c. Batch data delivery: consolidation and reformatting
- d. File format adjustments (Linstedt, 2011)

**EDW – Data Vault.** “The DVM follows all definitions of data warehouse except one (Inmon’s definition): DV is functionally based, not subject-oriented – meaning that the business keys are horizontal in nature and provide visibility across all lines of business” (Inmon, 1992). The EDW, or core historical data repository, contains the DVM tables. EDW holds historical data at a granular level (raw data set). The DV consists of Hubs, Links and Satellite. Unlike other existing data warehouses in present day, referential integrity is complete across the DVM. DVM has a greatly normalized architecture, but some satellites can be de-normalized to a degree, under specific circumstances.

The DV architecture follows the 3 1/2 normal form. The business keys (i.e. Hubs entity) follow the 6th normal form, while the load date and record source are in the 3rd normal form. The DVM must follow the lowest possible grain. Hubs and Links in the DVM provide the back-bone structure of the model, and satellites provide the context and are applied to Hubs and Links (Linstedt, 1992).

**Metrics Vault.** Metrics Vault is a module for capturing technical metrics about the load process, completion rates, amount of data moved, loading time-lines, growth of tables, indexes and files. This DV captures the technical metadata for the processes. By capturing growth rate actuals, along with run-times, insert update numbers and row counts. – projecting the future data storage requirements are created and managed. This helps to monitor needs and budgets by 6
months to 1 year in advanced, for future enhancements. This vault can also be crafted to include CPU utilization, RAM access; I/O throughput and I/O wait times information. The additional information in metric vault provides a consistent and concise view of utilization of the system, in conjunction with growing data sets and the hot spots on disk. From the metrics, a nearly complete technical management dashboard can be present to monitor the EDW effort (Linstedt, 1992).

**Meta Vault.** The Meta Vault consists of business metadata (definitions/taxonomies) and physical data model attribute names, functions and technically applied business rules that ETL follows to intercept the data. This vault also allows businesses to produce, maintain and deliver metadata across the board from within their EDW/Bi solution set. It is, in fact, one form of operational Data Warehouse. The Vault contains metadata for the staging area, EDW Data Vault, Report collections, Data marts and Metrics vault areas. Metadata is defined through business, IT and process technologies (Linstedt, 1992).

**Report Collection.** The Report Collection can be defined as a flat-wide, de-normalized structure, utilized for faster reporting or flat file output; it can also be used for data mining/text mining tools. There are some forms of data mart the end user can directly access. Report collection also provides the business users with pre-computed totals, allowing fast filtering against patterns of rows that are “out of normal Zone” (Linstedt, 1992).

**Data Marts.** “Data Marts” are defined as any point at which generic users directly access the data for ad hoc reporting, or drill down analysis, and access the structures directly. This may or may not be star schema and similarly, it can or cannot be normalized or de-normalized tables. Data Marts can be virtualized. A form of Data Marts can be an Excel spreadsheet that can directly communicate to the Data Vault through an interactive metadata layer. Direct communication between the user, the metadata management, and the DV are the beginnings of an operational DW. For auditing and accountability, the data is kept in two physical layers: corporate marts and error marts. Corporate marts serve as standard DM, where data meets soft business rules. Error marts serve as a landing zone for “bad data”, i.e. data that does not meet the business rules (error logs) (Inmon, 1992; Kimball, 1996).

**Business Data Vault.** BDV is a new concept that is not depicted in the architectural diagram, but is mentioned in detail here in this paper, due to the benefits of having embedded business data. Business users are worried about the benefits like flexibility, scalability and adaptability of DVM. Downstream of the raw data vault, between the DV and the Data Marts, modelers are building a new storage called “Business Data Vault”.

Business Data Vault is a concept of grouping specific tables using DVM concepts, but not all the raw DVM rules. A BDV can be grouped as tables inside the raw DV, or it can be a completely separate data store. Either way, the data residing in BDV can be altered, cleaned and changed to meet the rules of the business. We can dual-purpose the BDV and apply master data rules as well, thus making the business DV a beginning for a Master Data system.

The Business Data Vault will have all the business data, altered data, aggregated and cleansed information. Developers are executing the business transformations, assigning more metadata and then releasing the data information needed in the marts. The BDV is an extra copy of the information; however, it is paired with metadata and its transformation is necessary for making virtual cubes and high speed delivery possible. The arguments received from the business are the data used on the financial and management reports and these data have to be accounted and audited. Thus, the second copy of the data is very important (Linstedt, 1992).
Operational Data Vault. The Operational Data Vault (ODV) is a part DW and part on-line transactional data store. It stores all the changes to data as inserts; however it also offers to update and modify access to operational applications residing on top of the data warehouse. The nature of raw DV is going to be continuously changing to accommodate operational data. There is a need to combine and consolidate operational data with the raw DV, which is being driven by master data initiative and business needs. The business always wants more historical data mixed with current transactions. In order to meet business needs, the data warehousing developer teams are loading operational data (i.e. real time loading) directly into the raw DV, thus, creating an ODV (Linstedt, 1992).

Dynamic Data Vault. The Dynamic Data Vault is an operational DV with a dynamic adaptation of structure. In other terms - words, tables, columns, indexes and keys are all subject to change automatically. In order to achieve this state, it requires constant observant watch on the metadata, including but not limited to, incoming structures. This may include XSD, XML, staging tables or metadata that describe the structure of incoming data. The dynamic nature of DV means new attributes can be added to satellites while new Links and Hubs can be formed on the fly. ETL loading script can be adjusted spontaneously, and SQL views will also inherit certain changes. Scheduled emails will be sent out to the team (IT staff team) for review after changes are made to the model automatically (Linstedt, 1992).

Concepts of Data Vault

This paper focuses on DVM in a traditional method, which is model driven development with an appropriate platform independent Meta model and few Meta concepts that have clear explanations. The Meta model is used to explain semantics of complex data warehouse systems and to mediate various transformations.

All DV concepts are in a real context of time (generally bi-temporal), with their location being local for a system subject, based on resolution (standard, law and contract). The primary Conceptual Data Vault, "p-concept", is defined as depiction of interest, whether it is real or abstract (as we know, only observable concepts make sense for real data warehouse), and it exists as either a Hub or as a Link.

Hub Entities. The “Hubs” are defined as “unique list of business keys.” They are supported with additional technical Meta data elements, such as load date time stamp, record source, sequence number and last seen date (Linstedt, 2011, p. 34). These listed keys can be used as a composite (made up of more than one field), intelligent key (smart keys), or sequential keys (which are sequential in nature). This key property differentiates entity instances and may serve as the truth about the entity. All other properties about entities are changeable over time, including their connection to other entities.

The function of a Hub is to track a business key when it arrives in the data warehousing loads and the parent source it arrives from. Hub is a key recording device in the DVM. The business keys in a Hub should be defined at the same granularity. For example, title in a Library Management System is an individual granularity and forms a Hub.

| Title_Hub |
| ISBN |
Link Entities. A link represents relation between concepts. It can be defined as “an intersection of business keys”. It is said to contain the surrogate key ID that represents a particular Hub and Links parent business keys (Linstedt, 2011, page 48). A Link is present when there are a minimum of two parents table. A table’s grain is defined by the number of parent keys they contain. Another type of Link is expressed as a relationship in UML (Also a “weak entity” in ER relationship notation).

A Link provides flexibility for a DV model to change its structure over time and mutability of the model without losing its history of success and long term viability of the EDW. The Model itself can be modified, morphed and adapted.

Satellite Entities. A Link or Hub context is delivered by satellite. A satellite is used to associate a Hub (a Link) with an (data model) attribute. An instance of a Hub should always be related to at least one satellite instance (Linstedt, 2011, page 75). This condition satisfies the 6NF (Date), but they are relaxed in DVMs. Hubs can be related to more than one satellite. The Key can be a composite key if it is composed of more than one value. Any Key instances identifies only one Hub. The term “value” stands for datum (Single data value) in traditional terms.

Entity and Attributes pattern. Every entity is converted into Hub that is related to the satellite, which contains source attributes. The satellite shown follows a 6NF expectation. Each of these attributes needs a separate satellite; this assures a principle of independent updates by addition. Physical DVMs with inseparable multiple attributes will use a single Satellite, instead of one each (Jovanović V., Bojičić I, 2012).

Association Pattern. Associations are changed to Link entity (M : M cardinalities), regardless of association cardinalities in the source model (Jovanović V., Bojičić I, 2012).
Aggregation Pattern. Both classes become Hubs, and aggregation is converted to Link concepts. All the attributes are given a satellite (Jovanović V., Bojičić I, 2012).

Association Class Pattern. All classes involved in association are converted/transformed to Hubs, while the Link that relates these Hubs gets Satellite(s) with other additional attributes. Note that an Association Class that participates in the association of their own would be handled as a Hub connected to the Link that represents the original Association Class (Jovanović V., Bojičić I, 2012).

Generalization Pattern. All specialized relations are converted to a corresponding Link (Jovanović V., Bojičić I, 2012).

Recursive Association Pattern. Links are related to Hubs two times (related to itself) one for each role (Jovanović V., Bojičić I, 2012).

It is recommended by senior modelers to use Identification Definition (IDef1X) standard notation (instead of UML notation) for efficient and effective reengineering of physical Relational Database models, prior to C-DV modeling, due to IDef1X’s smallest rational distance from the Relational Model.

Data Vault Evolution, DV 2.0

Even DV has been addressing the issues with DW in the last 20 years very well. There is a need to improve DV 1.0 because of new material and changes happening recently, which simply will not apply well in existing standards of DV 1.0. DV 2.0 has a bigger focus on implementation; addresses many performance issues of data models, Big Data, and unstructured data; works well with big data systems, such as Hadoop and HPCC; and works well with MPP databases like Teradata; and works well with issues that come from NoSQL databases, like Cassandra, MongoDB, etc (Pande, 2012).

From the implementation perspective, DV 2.0 covers not only modelling which is part of DV 1.0 or the “Integration Layer”, but also covers the end-to-end solution architecture and methodologies such as Agile, TQM, etc. From a flexible scalable Hub & Spoke system, which includes hashing, loading patters, and housekeeping, DV 2.0 has a DV architecture, which is a multi-tier scalable low impact system integrated with DV Methodology, which uses consistent repeatable and pattern-based methodologies such as Agile, CMM, TQM etc.

The changes in the Architecting component impact how we develop ETL for DV 2.0. ETL will use a hash value of a business key and not an integer value. The hash values will be generated the same way everywhere in the system and will increase the performance while reducing complexity. The key lookup tables will be removed for all HUB, LNK, SAT and LSAT templates. From the information delivery perspective, there is no difference in the delivery of the information, but it allows you the option to use ETL tools to create “Information Mart” (Vos, 2014).

Conclusions

This research presented the initial physical design stage of Data Vault types of Enterprise Data Warehouse (EDW) that is integrated DW as systems of records that are not open to the end
user reporting. It is handled on the basis of incremental expansion of data warehouses that adds a new data source in sets or one at a time. This makes the Data Warehouse more scalable and flexible to expand (Dragoljub, K., Vladan, J. & Zoran, M., 2013). The algorithm uses metadata and rules for designing the direct approach to physical DV design mainly transactional data source. The relationship between entities in transactional source systems and rules for the development of a data warehouse based on the DV concept is crucial for the physical design of automated the data warehouse model. The concept of metadata is present in the physical model that can be used for the design of individual data warehouses, and this becomes the basis for the development of a tool. The most important contribution in this paper is the realization of DV directly from RDBMS schemes (evolution of Data Warehouse to DV).

Traditional ways of integration use either snipping of data from the source or other forms of derivation that is consolidation that requires a lot of intervention by experienced modelers (due to creativity and rich semantic transformations). DV provides a unique ability to integrate data incrementally by adding links (essentially 1:1 mappings) between initial and added hubs, while preserving all data in satellites, links and hubs without any reconstruction and deletion.

There is ongoing research about detailed specifications of dynamic expansion of Hub, Links and Satellites. Progress of work is focused on code generation for the initial loading of created Data Vault EDW, as well as code generation for the DV updating with new values and updates to the update-code. The PDV approach is based on available relational schema and this satisfies metadata requirements stated earlier. Any indirect DV design driven by concepts or by logical data model, even when it is supported by some automation, is less flexible than direct PDV. Additionally, it also increases risk of losing data from the source, potentially invalidating the central DV purpose – to retain a system of unaltered records.

The DV type DW is a solution for integrating, storing and protecting data. Moreover, it is either intended, or suitable, for intensive queries or reporting. This is why the DW architecture with a DV layer (that stores historical data) also contains a data marts layer that uses star schemas for reporting data. The dimensions of the star schema result from the Hubs and Satellites tables, and the fact tables result from a Satellites and Link tables.

The next generation topic of research is addressing the output areas Data Marts (DM) with the following steps:

1. Create a model of DM’s and DM’s materialization code
2. Create metadata for Analytics Tracking (scorecards/dashboards)
3. Standard reporting

The process of designing an EDW based on the DV model can be formalized and generalized to a great extent, based on the automated physical model for structure; semi-structured and simple unstructured data sources, that include transactional databases. The direct approach integrates elements of the physical design of EDW based on DV model as a system of records. This paper also illustrated the development of a tool for automation design for DV based EDW.

References:

References available upon request