Topofminds

Top of Minds Report series

Information Integration – Hyper Agile Design Pattern for Data Warehouse

Introduction

In the Report "ToM Report Series on Information Integration – IT Support of Business Agility" we looked at the idea of building flexible information integration solutions, on a high level, that could

- Quickly adapt to today's fast moving business environment
- "Absorb" new business information requirements
- Give the business users the power of how the information should be translated into a common business information view
- Ability to set up rules for how the information should be used by business processes.

This report will focus on how to design a Hyper Agile Data Warehouse.

We will look at different design patterns and how to combine them to create a Hyper Agile Design Pattern. This pattern can be used for many different information integration solutions, Master Data, Data Warehouse, Operational Information Integration etc.

Data Vault

Data Vault is one of the most flexible Data Warehouse paradigms of today, invented by Dan Linstedt. Data Vault is both a Data Warehouse methodology and a design pattern. In this report we will have a quick look at the modelling design pattern.

About the author

Patrik Lager is a senior specialist at Top of Minds. Patrik is specialized in data warehousing architecture, information modeling, data modeling and ETL design. He has a long and vast experience from working within bank & finance area and also telecom.
Mr. Lager holds a BSc in Computer Science from Linköping University. He is a member of the Data Vault Standardization Institute.

Comments on this white paper can be sent to patrik.lager@topofminds.se or Twitter @PatrikLager

About Top of Minds

Top of Minds is a specialized company that offers services in the data warehousing and business intelligence area. We are premier partner in the Nordic countries on the data warehouse development using an agile approach where we use Data Vault to increase the benefits of the projects we participate in. We are a company with great focus on competence, our expertise and our clients' expertise and we are constantly working to spread our knowledge.

See www.topofminds.se for further

Data Vault Concept and Structures

The core of the Data Vault modelling technique is the identification of the core business keys, such as Product or Customer. All business keys, and only the business keys, are stored in a specific table structure called a Hub. The business keys might have certain relationships, such as that between an Order and a Customer. All business key relations, and only the relations, are stored in another table structure called a Link. None of these, Hub or Link, contain any history. Data Vault captures the existence of the Business key in the Hub or the Link between business keys once and after that never changes it.

All descriptive data, for instance the name, address and telephone of a Customer or the date and shipping information of an Order is stored in a third table structure called a Satellite. This image, see Figure 1: A simple Data Vault Model, Figure 1 illustrates a typical small Data Vault model.

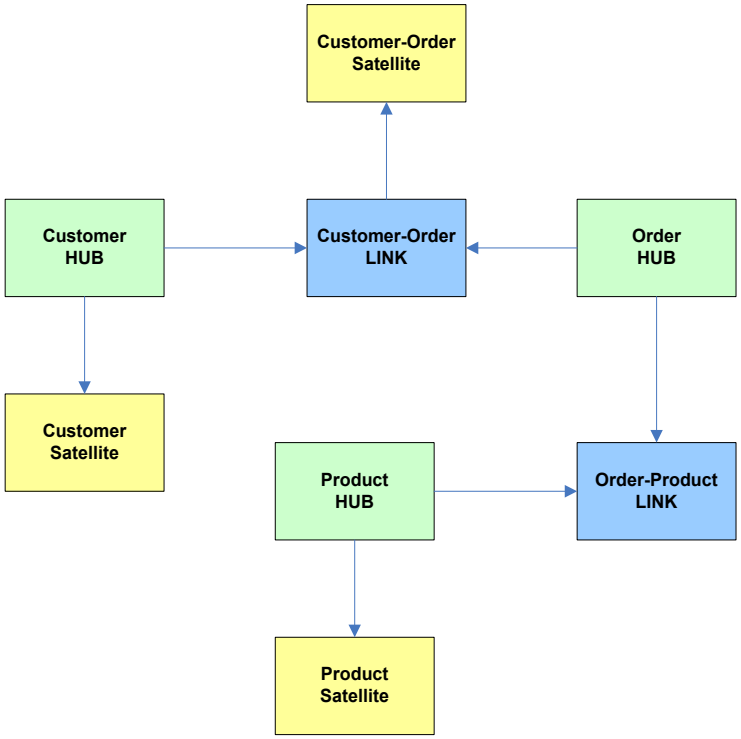

Figure 1: A simple Data Vault Model

Data Vault agile design pattern

One of the major strengths of Data Vault is its agile structure; which makes it easy to create a modular code base. Since the design pattern is very clean with few constructs, it is very easy to repeat and reuse code for loading data into the tables. For more information about loading strategies and their reusability see “ToM Report Series on Agile DW - Using Informatica PowerCenter to automate your Data Vault”.

Data Vault has created a model/code pattern for scalability. The ETL code for the Hub, Link and Satellite is self-contained. That way the ETL code becomes modular and the changes of adding new business keys or relationships or attributes are contained within its specific module of ETL code, not affecting other parts of the code.

By breaking out the descriptive data, which is most prone to change, from the business key and put that into the Satellite, Data Vault “protects” the ETL code of Hubs and Links from changes.

When adding new information/data to a business key, you can either add a new satellite or add new attribute to the existing Satellite, depending on the nature of the information added.

So even there, Data Vault gives the choice of keeping the Satellite code unaffected, if deciding to add a new Satellite to hold the new information instead of adding it to the existing Satellite.

Designing the Hyper Agile DW

Hub and Link

The Hub and Link pattern from Data Vault has just the right modular building blocks to support agile development. This creates a model built on the business keys used in the company’s business processes. That will help the business to understand what information they are working with at the Meta Data layer.

Satellite

A normal Data Vault Satellite is prone to code and table changes. Each time you want to add more attributes to a business key, you either have to create a new Satellite or change the existing one. This is not acceptable in a Hyper Agile solution.

Name Value Pairs

Name Value Pairs, Attribute Value Pairs or Key Value Pairs, will henceforth be called Name Value Pairs, is a technique that creates a structure that, instead of being agile, is absorbing new attributes. That means that new information/attributes that are added to a Business key require no change in table or ETL code, it simply “absorbs” it.

A Named Value Pair Satellite has instead of a unique column per each attribute, a generic column that holds the Attribute name and a generic column that holds the attributes value.

Let's look at the difference. We can see here, see Figure 2: Normal Satellite Table, that for each Business key and Load Date, which is the Primary Key, we have one row.

Normal Satellite					
Business Surrogate Key	**Load Dts**	First Name	Last Name	Adress	Age
1	2011-01-01	Patrik	Lager	Ashenroad	42
2	2011-01-01	Bengt	Bengtsson	Knownroad	35

Figure 2: Normal Satellite Table

The Name Value Pair Satellite, see Figure 3: Name Value Pairs Satellite, has another structure. For each Business Surrogate Key, Load Date and Name (Attribute), which is the Primary Key, we have one row.

Name Value Pair Satellite			
Business Surrogate key	**Load Dts**	**Name**	Value
1	2011-01-01	First Name	Patrik
1	2011-01-01	Last Name	Lager
1	2011-01-01	Adress	Ashenroad
1	2011-01-01	Age	42
2	2011-01-01	First Name	Bengt
2	2011-01-01	Last Name	Bengtsson
2	2011-01-01	Adress	Knownroad
2	2011-01-01	Age	35

Figure 3: Name Value Pairs Satellite

Name Value Pairs can be designed in many different ways in a Data Vault model, for this Report we simply use a satellite construct. Below, see Figure 4: Hub and Name Value Pairs Satellite, is a Customer HUB with its Named Value Pairs Satellite.

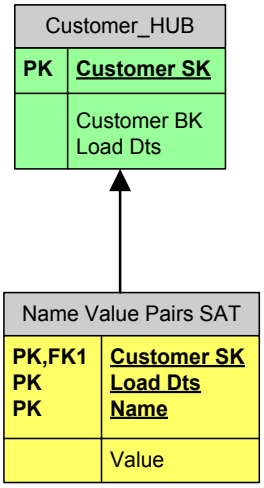

Figure 4: Hub and Name Value Pairs Satellite

There are now the building blocks for our data model where we will store the source data.

The Architectural Layers

There are some aspects that are important to understand when designing this kind of solution for a Data Warehouse. So let's look at the different architectural layers and their design rules.

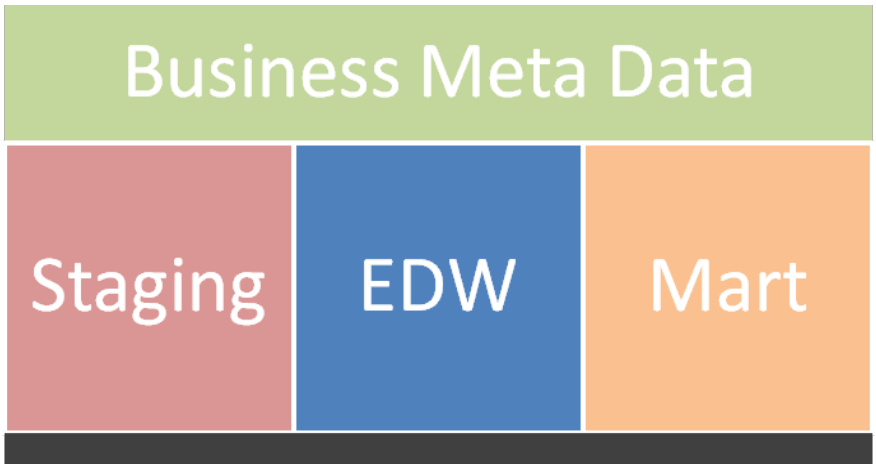

Figure 5: Architectural Layer Overview

Staging Layer

The staging area is a temporary area where data is captured. There are three main purposes of the Staging Area.

- Transform the data structures to Name Value Pairs
- Feed the Business Meta Data layer with attributes from the source systems
- Conform the data model to the EDW layer.

The data is source unique and has no information integration.

EDW Layer

The EDW model is a Data Vault skeleton model with Hub and Links and satellites of Name Value Pairs design.

The main purpose of the EDW layer is to have all the source systems data to be co-located in a common data model. There are some important aspects that differ from a “normal” Data Vault or other Data Warehouse design patterns

- The data is held source unique, there are no business key integration in the hubs.
- The attribute names are the same as in the source system, no attribute consolidation
- The attribute values are held in the source systems raw format, there is no value consolidation
- The only integration is that the information of different source systems is co-located into the same Data Structures.

Example: Key Integration

If a two source systems sends customer information, their customer key will be stored in the same hub, see Figure 6: Customer Hub Table. If they have the same business key, the solution will still store them twice in the hub, each with its unique Customer Surrogate Key.

Customer Hub

Surrogate Key	Business Key	Load_Dts	Source System
1	Patrik L	2010-01-01	Customer system 1
2	Patrik L	2010-01-01	Customer system 2

Figure 6: Customer Hub Table

Example: Attribute Integration

From an Attribute integration perspective; see Figure 7: Customer Name Value Pairs Satellite 1, if both of these systems send information about the customer’s first name, the solution will store that twice, once per each Customer Surrogate Key and with the unique attribute name of the source system.

Customer Sat

Surrogate Key	Load_Dts	Name	Value	Source System
1	2010-01-01	F_NM	Patrik	Customer system 1
2	2010-01-01	First_nm	Patrik	Customer system 2

Figure 7: Customer Name Value Pairs Satellite 1

Example: Value/Code Integration

From a Value/Code integration perspective; see Figure 8: Customer Name Value Pairs Satellite 2, If both of these systems send information about the customer’s country of birth and the business want to see country according to the standard “ISO 3166-1 alpha-2” the data that the source systems sends will be stored in the raw format even if it doesn’t follow the ISO standard.

Customer Sat				
Surrogate Key	**Load_Dts**	**Name**	Value	Source System
1	2010-01-01	Co_of_Birth	Sweden	Customer system 1
2	2010-01-01	Country_BRTH	sw	Customer system 2

Figure 8: Customer Name Value Pairs Satellite 2

All information integration work that normally happens in the ETL code when loading the data tables, are now manage through a Meta Data Layer. Which means that the business will get a “soft” information integration.

Business Meta Data Layer

The Business Meta Data layer has three main purposes.

- Hold the information of how all that data in the EDW is seen from an information integration point of view. That means that Business Key integration, Attribute Integration and Value Integration, they all happen in the Business Meta Data Layer, see Figure 9: Meta Data Layer.
- Hold the rules on how the Data Marts are to be loaded
- Hold Meta Data on the source system tables, attributes and source systems domain range values

All the Meta Data is historicized so changes in the Meta Data are captured.

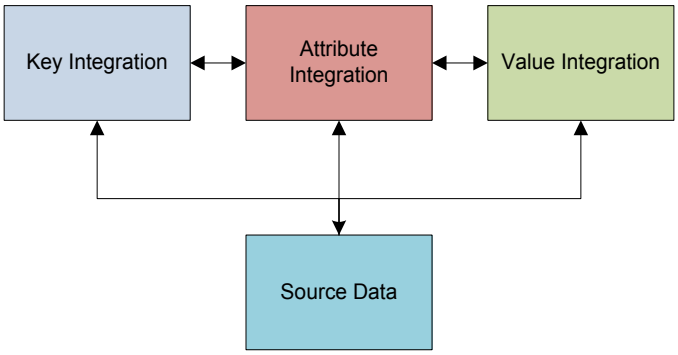

Figure 9: Meta Data Layer

Look at the Data through the Meta Data

When accessing the raw data through the Business Meta Data layer you get the Enterprise view of the data, as if it had been loaded and stored with key integration, attribute integration and value integration in the Data tables.

If a user want to know the Country Of Birth for one of our customer, a certain Patrik Lager.

Key Integration would show the user that the two rows for Business key “Patrik Lager” in the Customer Hub is really the same Customer. So only one row is returned for that Business Key

From an Attribute Integration point of view the user would see that the source unique attributes “Country_BRTH” and “Co_of_Birth” have been “mapped” to the same business attribute, “Country Of Birth”

The Value Integration part would show the user that the values “Sweden” and “SW” both point to the same domain value in “ISO 3166-1 alpha-2”, SE.

So the result of the query when translated by the Meta Data Layer would be Query Result 1

Query Result 1		
Customer	**Attribute**	Value
Patrik Lager	Country Of Birth	SE

So instead of storing the integrated result of the integration rules in the data tables, they are held in the Meta Data layer. Since the data itself is in its raw, source unique format in the data table, the business can change the meaning of an Attribute, or change a key integration etc in real time, without reloading the data, just by changing the Business Meta Data.

If a business user rather would see that the two source system attributes “Country_BRTH” and “Co_of_Birth” should be integrated as the business attribute “Country Of Residence”. The user would only need to change the Meta Data relation between the source systems attributes so they pointed at “Country Of Residence” instead of “Country Of Birth”.

When asking for Customer “Patrik Lagers” Country of Residence the user would get Query Result 2

Query Result 2		
Customer	**Attribute**	Value
Patrik Lager	Country Of Residence	SE

The underlying data in the data table has not changed, only the business interpretation of it without reloading anything.

At the same time, since the Meta Data holds the history of its rules, you can position yourself by date and time to see how the integration looked before it was changed.

Mart Layer

The Mart Layer is an instantiation of the Business Meta Data Layer. Here the data is stored according to the rules set up in the Business Meta Data Layer. What modelling technique you use in the Mart Layer is depending on need. Each Mart can have its unique modelling technique. It can be Dimensional modelling, Data Vault, 3NF etc. Whatever suits the usage of the information in that specific Mart.

This layer will not be as flexible as the EDW layer, since we have loaded the data according to the rules instead of keeping them “soft”. The reason for this is simply query performance.

It might be in the future there will be DBMS engines that are able to have query performance without the need of instantiation of the rules. Then we will be even more flexible.

Conclusion

The Hyper Agile Design Pattern is a Meta Data driven pattern focusing on the flexibility of absorbing new attributes into the Data Warehouse without any need of reengineering code or tables up to the EDW Layer.

The solution also give the ability to change the meaning of business key integration, attribute integration and value integration without and reengineering of code or reloading of data and still have the ability to see how the information was integrated at a earlier date.

This will save the business a lot of time and money and at the same time have the ability to set up the information integration rules on their own if they so wish.

If you found this Report interesting and want to hear more, please contact Top of Minds.