Topofminds

Top of Minds Report series

Data Warehouse – The six levels of integration

Recommended reading

Before reading this report it is recommended to read ToM Report Series on Data Warehouse – Definitions for Integration that can be downloaded at http://topofminds.se/wp/aktuellt/publicerat/. Integration is based on definitions and much of what is said in this report is based on what the report above came up with.

Background

Sometimes organizations has only a few source systems in its IT landscape and others have 100+ systems with overlapping information areas from different departments, divisions and countries that has to move its data into the data warehouse. They all have one thing in common, the information has to be integrated to fulfil the purpose of a Data Warehouse, make information accessible and usable throughout the organization independently what system, process, department, division and country it was created.

Data Warehouses primary aspect is integration of information. The integration has two major parts. Data model as such, to represent the common view of information. This is both a logical and well as a physical representation. The other big part of integration is Semantic integration. Model and Semantic integration are very tightly intertwined and they are two sides of the Information Integration coin.

In this report we will look at information integration from a Data Warehouse perspective, and break down what we need and how to think when working with it.

Target audience

This paper turns to new as well as experienced people in the Data Warehouse community.

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 in working within the 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.

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.

Data Warehouse – what is it?

It is important to understand that the definitions I use for the report are–

• Data Warehouse; for storing integrated information to support Business decisions.

• Business Intelligence; usage of integrated information to take business decisions.

Focus on the primary aspect

Data Warehousing is about one thing and one thing only – Information Integration. Everything else is secondary. Information Integration needs one thing and one thing only - Definitions. Everything else is secondary.

Your Data Warehouse never gets better than your information integration and your information integration never gets better than your definitions. So according to this all Data Warehouse projects/programs/organisations should have their main focus on two things;

• create good definitions in concert with the business(if no other department does that)

• Use definitions to create the best information integration possible.

The question here is, do they? The answer is often no! If you don't believe me, check your own Data Warehouse project/program/organisation and count the number of ETL architects/designers/developer, DBA's etc you have and then count the number of people working with definition management and then the number of people that try to use these definitions to create the fundamental information integration for your Data Warehouse. You will probably see the same pattern as in Figure 1 below.

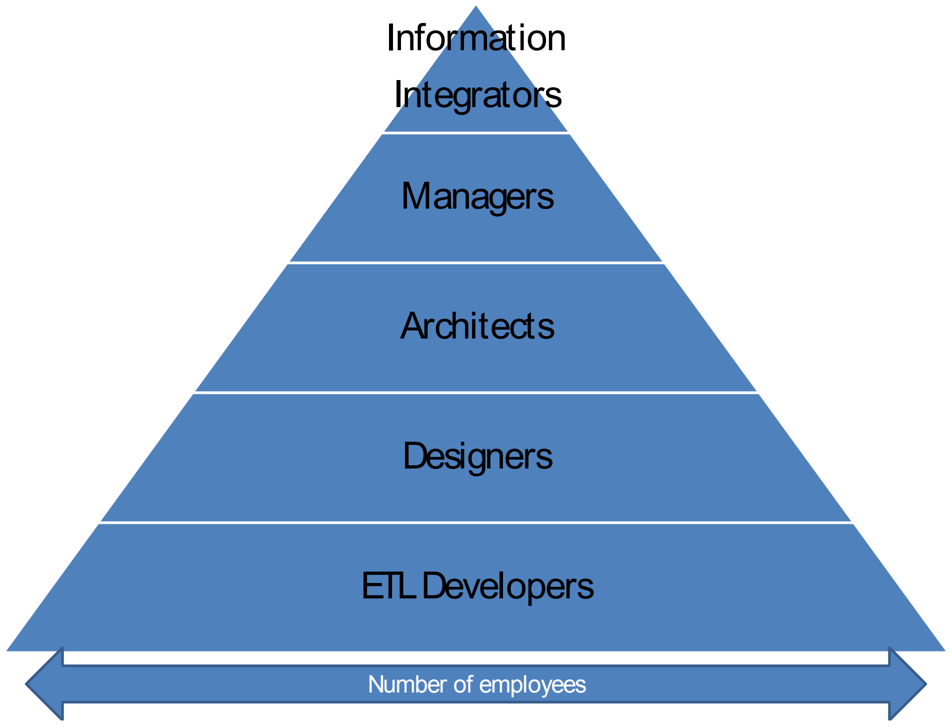

Figure 1: Number of employees in DW organization or/and projekt.

The point here is the pattern of number of people employed. The most difficult part of any Data Warehouse is the creation of definitions and the usage of these definitions to create good integration but the main focus in the Data Warehouse project/program/organisation is not always on that. While the Data Warehouse projects/programs/organisations focus on designing and building ETL code, managing hardware and software etc they tend to forget that the heart and soul of Data Warehousing is all about Information Integration and I believe that is the reason why we fail to fulfil the potential of Data Warehouse.

Why Information Integration is the primary aspect of a Data Warehouse

You could have manual information integration solution i.e.-a manual Data Warehouse using an old file cabinet. It would not be fast but it would fulfil the fundamental purpose of Data Warehouse; making information integrated for usage throughout an organisation. If the information is organized and stored according to the organisations definitions, different parts of an organization can (re)use the information, independently from where it was created.

On the other hand you could have

- The best ETL tool available that is able to automatically create ETL code based on source system data and loading millions of rows per second.
- The best Data base system with MPP and hybrid row column technology where data can be loaded and accessed with faster than lightning speed.
- Incredibly agile data modelling technique that supports incrementally growing data model with modular code approach.
- Super agile project management method with small iterative project phases where each phase is focused on creating business value.
- An Architectural design pattern, which is compliant, auditable, real time, reusable and scalable etc.

You could have all that and still not build a Data Warehouse, because if you don't have information integrated according to definitions, it is not a Data Warehouse, and that is why all of those other things are secondary.

Don't get me wrong, we need all the other parts, ETL, Data Bases, Project methods etc. to create a usable solution in today's fast moving information driven business. These things are all there to create an environment where the implementation and usage of the integrated information becomes as fast and agile as possible, but they are not what creates information integration and as such they are not the primary aspect of a Data Warehouse as shown in Figure 2 below

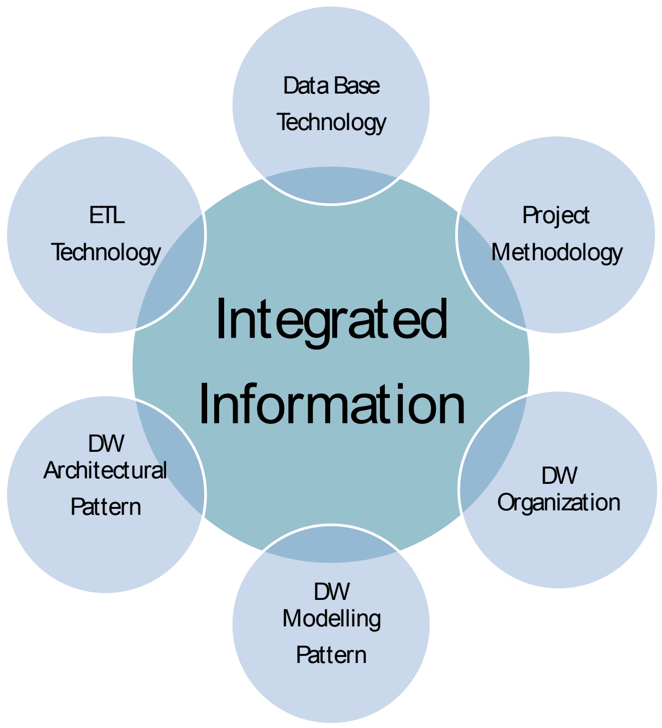

Figure 2: Data Warehouse Primary and Secondary Aspects

As the Integrated information is the primary aspect and the reason to build a Data Warehouse, it is important to understand that the other parts are support functions. Exactly as IT, HR and Finance is support functions for the Business in a company. What would be the reason too have the best IT, HR and Finance departments if the business didn't work? Same thing here, the secondary aspects of a Data Warehouse is support functions and without the fundamental Integrated Information they would not make any difference.

Information Integration

Information integration is about the ability of sharing and using information throughout an organisation independently what country/division/department/process/system created or updated the information.

Integration point

The integration point in the Data Warehouse, which holds the integrated information, logical or physical, can be in different layers of the Data Warehouse architecture. You could go from non-integrated raw data through integration points (Logical) to a data mart directly or integrate the data in a common layer (physical) that is reused by all Data Marts and multiple combinations of these two examples. What integration points that are set up where in the data warehouse architecture depend mainly on two things, how business users in the company needs its data for analysis and with what Data Warehouse architectural pattern you work with. The integration point can be on one single person definition level all the way up to enterprise definition level. The only thing that is true in the end is that we need to integrate the data accordingly to business user's definitions, so the user doesn't need to understand how each source system work.

Six levels of Information Integration

I will here explain the basic idea of six levels of information integration.

I have broken down Information Integration into six levels because I find that each level has its own purpose and challenges and that there are dependencies between the levels.

1. Common model integration
2. Key integration
3. Attribute integration
4. Consolidation integration
5. Code/Value integration
6. Format integration

Common Model Integration

When moving data from one or more source systems into the Data Warehouse we need transform the information into a common model. This Integration level can also be called "co-location of data", since the we at this stage are only concerned with data ending up in the Data Warehouse physical table that represent a defined Concept. Only data that fits the definition can implement the Concept and thus load the physical table that represent the Concept in the physical model.

So, here then is the first place we need our definitions. Without the definition of information concepts we can't decide which data record (instance) belongs to which physical table since the physical tables in the Data Warehouse has to represent certain concepts.

It might sound easy, but often the source systems physical data models are a challenge to transform to the common model and it takes a lot of time and effort to get it right since it is seldom a one-to- one relationship between the source model physical implementation and the common model physical implementation. Also the data in the source system will have modelling solutions that does not fit the Data Warehouse definitions. Does all the data that the source systems have in one of its tables match a specific Concept in Data Warehouse? If not, what data of different Concepts resides in the source system table and how should it be broken out and loaded into the physical tables in the Data Warehouse that represent the common Concepts?

It is seldom as easy as –"this is a system dealing with the XYZ concept and therefore we can load all data from that system into the Data Warehouse table XYZ". Careful examination of the data will probably reveal that it contain information about other common concepts and relations to other common concepts. These things have to be analysed and broken out into the Data Warehouse model according to the Data Warehouse definition of the concept. Otherwise the Data Warehouse will contain data from different source systems that do not match the Data Warehouse concepts definition and the Data Warehouse will fail its purpose.

With the help of good definitions and dedicated people, a well-defined information integration process can detect anomalies in the data and you will have greater success in creating good information integration at common model level.

Key Integration

In the first step, Common model integration, we have decided in what physical table certain data records (instances) from multiple sources should reside. If we have overlapping source system information how can we know that we don't have duplicates that represents the same instance in the information concept?

You could use the Values in the Characteristics of an Information Concept trying to decide if they are the same, if you have a table for "Person" then, two instances in that table having the same address, name, birthday etc, could be the same instance, but you can't be 100% sure. You could also check the business key of the source systems instances and if they match e.g. if each of the source systems instance are identified with the value 12AB346, it might be the same instance, but you can't be 100% sure.

Key integration is hard work if it is to be done in the Data Warehouse, which it is not recommended, since that is one of the key features of the Master Data. Sadly though, most of the time the Master Data does not exist, is out of date or does not have your specific source system in scope yet. Thus even with a Master Data function you might end up doing it anyway.

To approach this challenge you need to understand the following things to make decisions on Key Integration

- Are the Information Concept instances shared and moves in the Process-landscape?
- Are the Information Concept instances shared and move in the IT-landscape?
- Does changes of Life Cycle or other specific events of the Information Concept instances triggers movement in the Process/IT landscape?
- Does the movement of the instance imply that it changes what Information Concept it belong to?
- Does the movement of the instance changes its business key?

As you can see, to manage key integration with good quality you need a certain amount of knowledge about processes, the IT landscape and the life cycle of Information Concept instances. In a larger organisation this information might be very hard to get but it is necessary if you are going to implement key integration.

Remember that incorrect key integration can do as much damage to your information as not doing it at all.

Let me give you two simple examples from real life, where key integration could have gone wrong if the Information Integration team didn't have the knowledge I described above.

1. Two source systems send information about Deposit Accounts in a bank and the Data Warehouse load all the Deposit Accounts into the Deposit Account table. If the Information Integration team had not gathered the information describe above they might had decided to integrate on Account Id from the two systems. But since they did, they knew that these two account systems never share account data with each other. The caveat is that both systems use the same type of number series, digits 1 to 99999999 million, to set their Account Number. The danger that these source systems will send the same Account Number as business key is very real, even when the instances are unique for each source system. The result will be disastrous when we start write the information

from one account to another because of the uncontrolled integration on the business key.

2. Another example where two source system also sending Deposit Accounts in a Bank and the Data Warehouse loads all the Deposit Accounts into the Deposit Account table. The information integration team had gathered the information described above and decides to integrate data on business key. This time it is correct because these systems are sharing the accounts between them depending on the accounts life cycle status. Source system 1 holds information about the account when its life cycle status is "Normal" but when the account changes its life cycle status from to "Overdrawn" Source system 2 takes over management of the account but still maintains the same account number. The Data Warehouse would had duplicates of the same Deposit Account if they had decided not integrate these accounts.

Just because two source systems does not share and move instances for a specific Concept in the Data Warehouse the same rule can't be automatically applied to all source systems loading data into that Concept. Each key integration rule has to be carefully considered for each and every source system you load into the same table.
These were two rather simple real life examples of the importance to treat the challenge of key integration with the outmost respect.

Attribute Integration

When the Common Model Integration and Key Integration are done, we know what data should be loaded into which table and, if possible, which rules for Key Integration we need to apply to remove duplicates. The next step is to integrate information on Attribute level, which is the Definition of Characteristics as a Concept.

Where the definitions of higher level Concepts, such as Product, Contracts, Order etc can have a more "generic" flavour in their definitions since most of the time they do not need to describe what value it can hold, Characteristics on the other hand often have to describe in detail what the value it holds will be composed of. Here the Data Warehouse basic rule of atomic data comes into play.

The need for atomic data comes from the need of flexibility. Where one department want an aggregated level of data and another department needs less aggregation on the data. Data on atomic level of definition sees to that the Data Warehouse always can deliver the needed data.

The danger here is to believe that the "raw" data from a source system fits the definition of your DW attributes. On a high level this might be the case, but there are always reasons to look closer on the raw data from the source system to determine if it really fits the definition of the DW attribute and thus can load its values into it.

When Data Warehouse definitions does not fit the raw source data

How many of us Data Warehouse professionals, me included, have not been forced by project time line or been fooled by imprecise definition to load data that is not atomic into the Data Warehouse and, after it's done, the business has asked for the atomic data? Known as "getting the eggs back out of the omelette", it can be impossible or very hard, and of course often dangerous from an information quality perspective. The only way to avoid this is to have precise and atomic definitions and applying them when creating the integration in the Data Warehouse.

Let's go back to the example of the Bank Account and with its attribute, Account Balance. The definition of the Account Balance in the Data Warehouse is: *Account Balance is the balance of the bank account at a specific point in time and the value it holds at that specific point in time represent the monetary value of the settled transactions and does not contain any accrued interest.*

Easy enough, but when the Information Integration team askes for the Account Balance from a source system, the source system have another definition of their Account Balance: Account Balance is the balance of the bank account at a specific point in time and the value it holds at that specific point in time represent the monetary value of the settled transactions **and its accrued interest.**

The account balance in the source system had the accrued interest as part of it balance value and that did not fit the Data Warehouse definition, so the raw data does not implement the concept. So somewhere on the way the raw data has to be transformed to fit the integration point (Common attribute). If we let the data flow through the integration point and mix balances with accrued interest and those without accrued interest, the support quality of business decision gets lower and there is a danger of inaccurate decisions.

Consolidation integration

When you put the Common Model Integration, Key Integration and Attribute Integration together you might end up with a scenario where two source systems send information about the same Data Warehouse attribute for the same instance of a Concept. An example could be that the same (Key Integration) customer (Common Model Integration) is sent from two different source systems, both sending information about the customers address (Attribute Integration).

You have two design choices here. Either you hold the data source system specific and make the choice on which data you are going to load into the reporting layer in a later stage or you set up rules of priority where one source system has the highest priority and is loaded first. If a value already exist in the Data Warehouse attribute and that value is from the source system with higher priority, then no other source system data will be loaded into that attribute for that instance.

Value Integration

When we know into which table we should load the data, when we have fixed duplication problem and integrated and consolidated all the source system attributes into the Data Warehouse attributes we move on to Value or Code integration

Value integration is applicable only for certain Characteristics, those whose definition describe what value range to be used to represent the values in a conformed way. If we have a Characteristic that represents the concept of Country, the Definition could be: “An autonomously governed area of the world where a country is a land area that has a government which is not subordinate to any other jurisdiction (super-national associations like the UN or European Union do not count). Countries are represented by the use of ISO-3166 Alpha-2 code set” I this case the definition also tells us that it uses the ISO-3166 Alpha-2 code set to represent countries.

Now imagine if we have two source systems that may send us information about in what Country the Bank Accounts has been registered. One source system send the value “SWE” for one Bank Account and the other system sends the value “United Kingdom” for another account. For the purpose of conformed way to access data the Data Warehouse uses ISO-3166 Alpha-2 code to integrate the data so it becomes source system independent. The first is set to “SE” and the other to “UK”.

Once again, it is important that the definition is as detailed as possible, like the example above where it describes what value range is to be used to represent the Characteristic’s values.

So what happens when we get a Country of account registration that holds the value “EU”? That is bad data quality and is not the responsibility of the Data Warehouse; the value simply does not get integrated since the “EU” does not exist as a country in our definition. The Data Warehouse does not stop the data from being stored in the Data Warehouse; it only reports that the value is not correct according to definition. Of course there are cases where the value range are so important that if the value sent from the source is wrong more drastic measures has to be used, such as stop the load process in the Data Warehouse.

Format Integration

Finally, we are at the final level of integration – Format Integration. It is important that the format integration is part of the definition for the Data Warehouse. How we represent a Characteristic’s value is important, both when calculations are done on values and when users are accessing the data. If it is a monetary value, does it have 2 or 3 decimals? How do we represent date etc? Decide how different formats should be represented, Date, Percentage, Monetary, Quantity etc. and implement them in the optimal way for your specific solution. Remember that the when users is accessing the integrated data of a specific data type, it should always have the same format, independent of source system.

The Scope of the Data Warehouse

There are many ways that the Data Warehouse industry tries to describe the scope of a Data Warehouse but it seldom takes ownership of the fundamental scope, acceptance of Definitions. You can't build an Enterprise Data Warehouse if your definitions don't have Enterprise wide acceptance (Read the chapter "Acceptance of Definitions" in the report ToM Report Series on Data Warehouse - Definitions for integration). Every time a source system is integrated into the Data Warehouse according to the definitions, the width of the organisations acceptance of the definition sets the scope of the reusability of the data and by that the scope of the Data Warehouse. If there is an Information Model based on definitions with division wide acceptance then you can build a divisionwide Data Warehouse, but you cannot build an Enterprise Data Warehouse based on those definitions.

Conclusion

A full Information Integration can only be done with definitions and the better the definition the better the information integration and that in the end will create a better Data Warehouse solution. All the six levels of integration have to be applied in the Data Warehouse to make it an information repository that the organisation can use, independent of where the information was created. The idea is that, when accessing the Data Warehouse, you do not need to understand what data looks like in the source a system; the data in the Data Warehouse is source system independent.