Designing and implementation of data warehouse

Introduction.

Business Intelligence has become a buzzword in recent years to support decision-making. Today we can find several database systems, which include data warehousing, online analytical processing (OLAP), and data mining technologies. Data warehousing provides efficient storage, maintenance, and retrieval of data. OLAP is a service that provides a way to create ad hoc queries against the data warehouse in order to answer important business questions. Data mining is a disciple comprising of several algorithms for discovering knowledge in a large bulk of data.

In order to build a data warehouse solution, we need to model a consistent architecture where the operational data will fit well in an integrated and enterprise-wide view as well as to take into consideration a handful of implementation strategies to provide a high-quality application. The design and implementation of a data warehouse solution sometimes is a very complex challenge in theory and practice. In this article, I will cover the main principles and techniques to design and implement a data warehouse, providing my own experience in such an overwhelming challenge.

This is the first article of a series of articles that I want to write in order to share my knowledge and experience in this subject matter.

The design of a data warehouse

Although many of the principles and techniques to design and implement a relational data model for an operational system are adaptable to data warehouse modeling, they cannot be carried across in a natural way. Thus, data warehouse modeling is a new discipline that is enhancing every day.

Operational and decision support systems

Most operational data is stored in relational database structures such as tables and their underlying relationships. They're highly normalized and optimized to support the business processes in the enterprise. This kind of schema is excellent for operational systems with several users working concurrently and executing transactions (where the most important criteria are data consistency), although it's not adequate for reporting applications (where the common

operation is data extraction and the most important criteria are the performance of the underlying query) using a huge amount of data by relatively very few users; because you have normally to join several tables in the execution of the query.

Decision support systems deal with huge historical data representing a time slice of the operational data. A data warehouse is the storage medium for decision support systems, and it requires periodic updates to load new data from operational data sources. Operations over data warehouse are characterized by read-only ad hoc queries (less predictable) over a high volume of data which might be integrated, aggregated, and summarized for decision support purposes. Table 1 summarizes the basic differences between operational (using relational database technologies) and decision support (using data warehouse technologies) systems.

Relational Database Systems	Data Warehouse Systems
Transaction oriented	Business process-oriented
Thousands of users concurrently	Few users
Generally small in size (Hundreds of MB up to GB)	Very large (Hundreds of GB up to several TB)
Current data	Historical data
Normalized data	De-normalized data
Continuous updates	Batch updates
Simple to complex queries	Very complex queries

Table 1

Dimensional modeling

Dimensional modeling is a technique to structure business dimensions and metrics, which are analyzed along with dimensions to execute high-performance queries. At a high level of interpretation, the data warehouse contains an integrated view of data that is derived from data in the operational systems supporting different business processes. In between the operational systems and the data warehouse, there is an important component known as the staging area. In this area, the operational data gets to be cleansed and transformed into a format suitable to be placed in the data warehouse storage.

A data warehouse database is a highly de-normalized structure with two main components: the first one is a central table, also known as a fact table, which

contains transactional data, and it is surrounded by the second type of components, known as the dimension tables which contain referential static data or master data. The conceptual relationship between these two main components is that the dimensions describe the facts. This specific data model of dimensions and facts tables is known as the dimensional model, and it can be graphically described as a star schema (a single fact table surrounded by dimensions) which specifies a multidimensional database. A multidimensional database can be built using conventional relational DBMS (it is referred to as ROLAP database) or specialized multidimensional DBMS optimized for such structures (referred to as a multidimensional database composed of cubes). A data model of a data warehouse is practically made up of data marts or a sub-set of star schemas, where each data mart is a single fact table surrounded by dimension tables containing data for different departments of functional areas.

Transformation from the enterprise data model to the dimensional model

The enterprise data model is the specification of the business entities as well as their underlying relationships, keys, attributes, subtypes, and business rules using the entity-relationship (ER) and normalization techniques. The enterprise data model is a global view of the data model of the database systems to be integrated into the data warehouse. This important model corresponds to the subject area supported by the underlying operational information systems. The degree of completion of the larger enterprise data model is of little concern to the development of the data warehouse.

Once the enterprise data model is specified, you can make several transformations in order to build the dimensional model of the data warehouse, such as,

- Removal of purely operational data
- Addition of appropriate derived data
- Transformation of relationships
- Creation of an array of data

It's remarkable to say that these transformations are a guideline for the design of a data warehouse, but their use is mainly determined by the business requirements of the decision-support systems.

Removal of purely operational data

This transformation deals with examining the enterprise data model and removing all data that is purely operational. They are not removed from the enterprise data models; they are simply not useful in the dimensional model. For

example, the definition of the entity Product (see Figure 1) contains a lot of operational attributes.

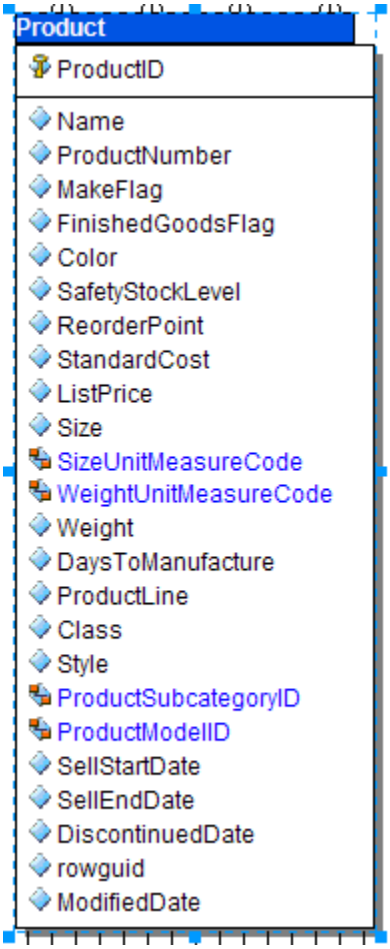


Figure 1

Now let's transform this definition into a new one that fits the dimensional model by removing the attributes: ProductNumber, MakeFlag, FinishedGoodsFlag, SafetyStockLevel, ReorderPoint, StandardCost, ListPrice, SizeUnitMeasureCode, WeightUnitMeasureCode, Weight, DaysToManufacture, ProductLine, Style, ProductSubcatogoryID, ProductModelID, SellStartDate, SellEndDate, DiscontinuedDate, rowguid and ModifiedDate. The final schema is shown in Figure 2.

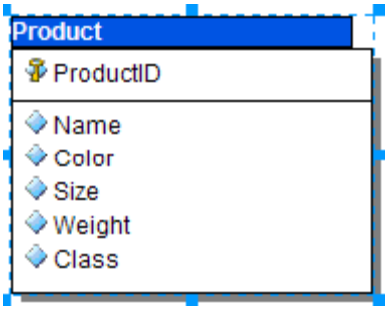


Figure 2

Addition of appropriate derived data

The next transformation is the addition of derived data (this is a de-normalization process), where it's mostly accessed and calculated in order to reduce the amount of processing required to get the data. Let's illustrate this transformation with an example of the Sales Order entity's data model (see Figure 3).

Error! Filename not specified.

Figure 3

Now let's transform this relational model in Figure 3 into a dimensional model (see Figure 4).

Error! Filename not specified.

Figure 4

Another case is when we have physically designed the Product subject into two relational tables, such as Product and ProductSubcategory tables in the AdventureWorks database (see Figure 5), and then we want to create a dimension for the Product subject. In this case, it's not descriptive to include the ProductSubcategoryID integer number; instead, it's better to include the name of the product subcategory (see Figure 6).

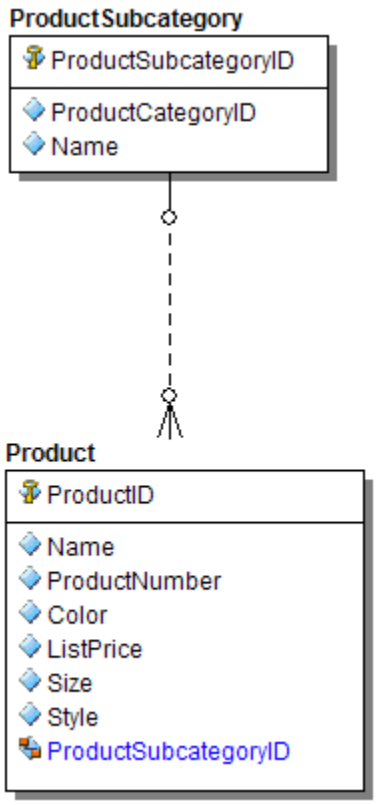


Figure 5

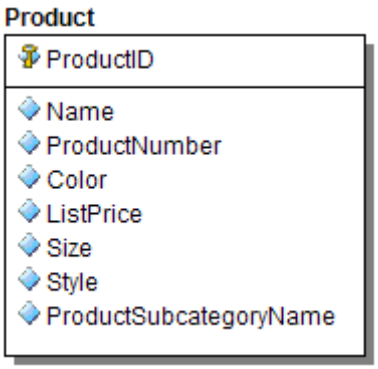


Figure 6

Another interesting case of this transformation is concerning the one-to-many relationship in the relational data model. Let's suppose that we have a non-identifying mandatory relationship between a Product and its Supplier. The underlying semantics dictates that a supplier may have many products, but a given product must have an associated supplier (see Figure 7).

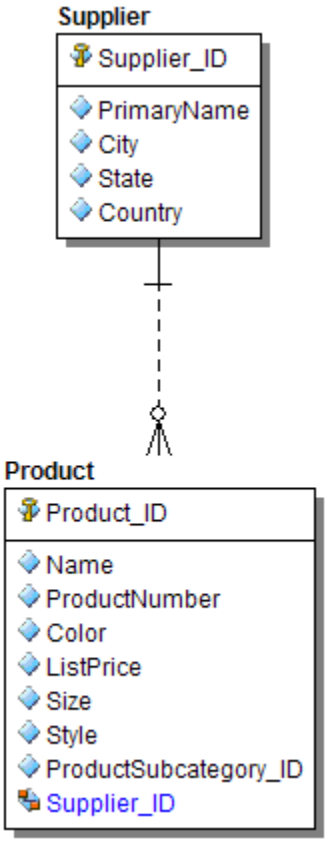


Figure 7

In order to turn this data model into a dimensional model, we design an entity that represents the Supplier and Product entities and the associated relationship. We can also apply the Removal of purely operational data techniques in order to design the most important attributes. This transformation is shown in Figure 8.

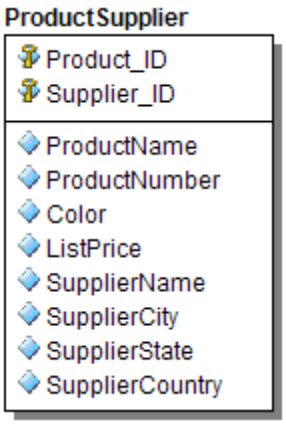


Figure 8

Creation of Arrays of Data

As the enterprise data model is usually normalized, then the repeating groups are not shown as part of the model. But under some conditions, the data warehouse, as an analytical tool, needs to represent these repeating groups.

Let's suppose that our data model designs the Budget business entity. The budget occurs on a month-by-month basis. In order to normalize this structure, we have the Budget and BudgetItem entities associated with an identifying relationship (see Figure 9).

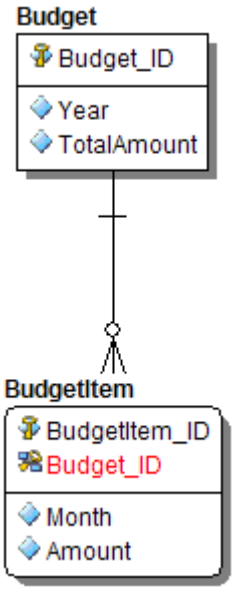


Figure 9

Now when we apply the transformation technique, we get a result in Figure 10.

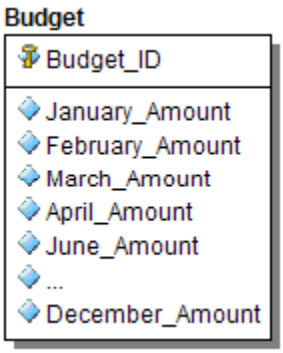


Figure 10

It's remarkable to say that this transformation is not a general-purpose option. There are several scenarios where this strategy may be applied, such as,

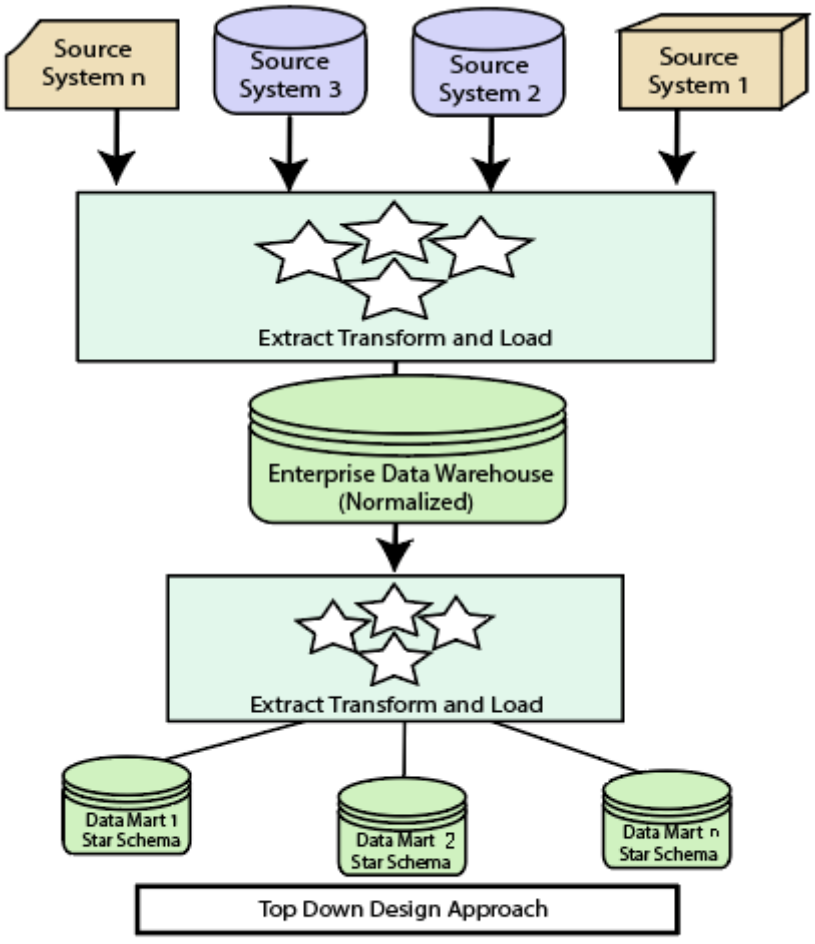
- When the number of occurrences of data is predictable.
- When the occurrence of data is relatively small.
- When the occurrences of data are frequently used together.

When the pattern of insertion and deletion is stable.

This technique is inflexible to changing departmental needs.

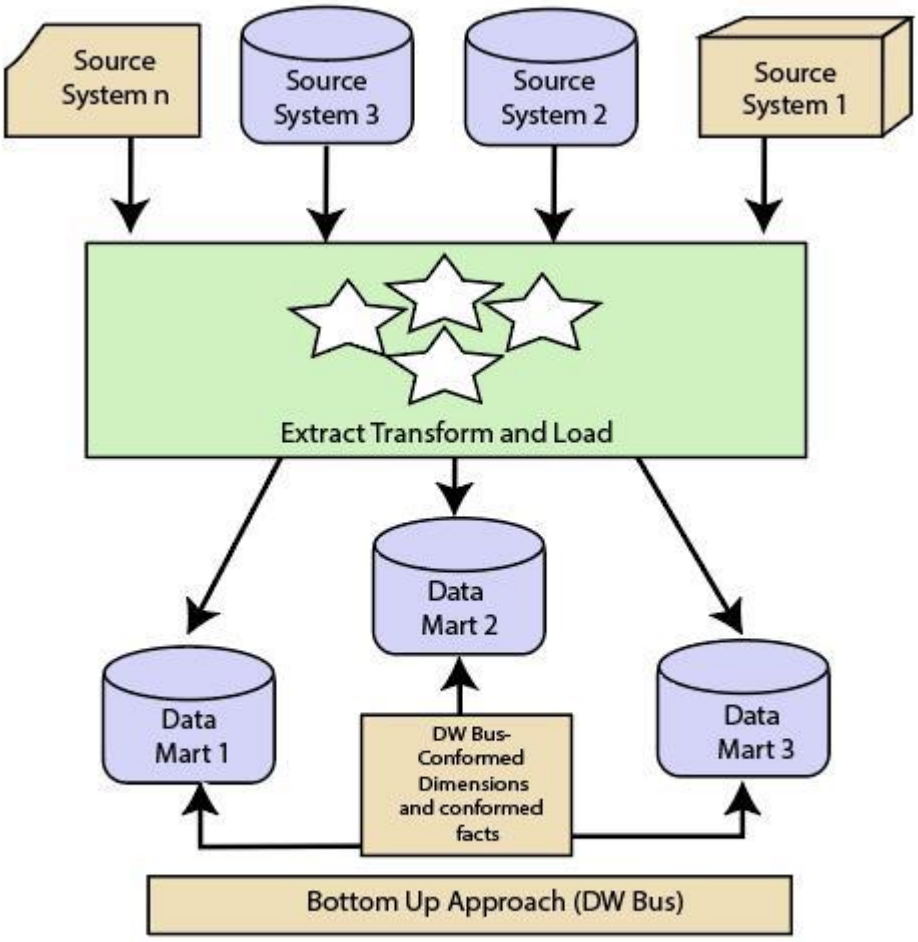
The cost of implementing the project is high.

Top down design approach



Top Down Design Approach

Bottom up design approach



Bottom Up Design Approach