Data Warehouse Design for Ease of Data Transformation

Ido Millet*
Penn State Erie, School of Business
Erie, PA 16563-1400
(814)-898-6262
ixm7@psu.edu

Diane H. Parente
Penn State Erie, School of Business
Erie, PA 16563-1400
(814)-898-6436
dhp3@psu.edu

John L. Fizel
Penn State Erie, School of Business
Erie, PA 16563-1400
(814)-898-6323
fzk@psu.edu

DMDW 2002: May 2002

*Corresponding author

Data Warehouse Design for Ease of Data Transformation

Abstract

Classical dimensional modeling aims for simple data warehouse designs in order to make the job of the end user and the DBMS easier. This paper describes a different approach aimed at simplifying data extraction, transformation, and loading tasks. Our design approach avoids and isolates changes to source data tables by surrounding them with helper-views. Each helper-view specializes in a single type of transformational task. The user is presented with a final view, which integrates the information from the source tables and the helper-views. We used this design approach in a data warehouse for analyzing half a million supplier bids submitted during Business-to-Business online procurement auctions.

Data Warehouse Design for Ease of Data Transformation

1. Introduction

At a previous DMDW conference, a claim was made that "problems like extraction, transformation and cleaning, which can take up to 80% of the time spent in the development of a data warehouse, seem to be ignored by the research community" [Vassiliadis 2000]. This paper describes a design approach, which simplifies the ETL (Extract, Transform, and Load) aspects of data warehouses. The aim is to share experience and insights from a project involving the design and use of a data warehouse for analyzing supplier bids submitted for thousands of business-to-business (B2B) online reverse auctions.

This project started when a large multinational company requested our help in analyzing their online reverse auction data. Our initial objective was to generate reports indicating best practices for conducting such auctions. For example, what auction characteristics (time window, number of invited suppliers, number of line items, overall auction value, auction type, etc.) lead to auction success? Sounds simple enough, but the project required much more data transformation work than we had expected.

2. The Transformation Imperative

Several complicating factors required that we use a very adaptable transformation mechanism. Measures of auction success were not clearly defined and the raw data required extensive and complex transformations to support even simple analysis. The data structure for the operational database was not designed with reporting needs in mind and suffered from missing data elements and design flaws at the source. We even had to cope with changes in

the structure of the source data mid-way into the project.

Many transformations required relatively simple formulas for establishing measures of auction success, converting currencies and time data, and conducting code lookups. More complex transformations were required for deriving non-trivial aggregate data such as the number of times each product and each supplier participated in previous auctions (at the time this particular auction was conducted). Other complex transformations were required to compute statistics such as skew measures for bid time distributions.

We quickly realized that much of the work and the real challenges were in the data transformation process. Based on past literature, this is quite common, yet past literature seems to have done its best to make this aspect of data warehousing as difficult as possible by emphasizing design for ease of use as opposed to design for ease of data transformation. Perhaps the time has come to reverse this trend and attend to the challenge of making it easier to transform and load data into data warehouses. Our approach to designing the data warehouse provides an example of such an approach.

3. An Adaptable Transformation Mechanism

Classical dimensional modeling aims for simple data warehouse designs (star schemas) in order to make the job of the end user (understanding and using the data) and the DBMS (processing of queries) easier [Kimball 1996]. In contrast, our approach was to design the data warehouse around the existing source data, using helper-views to provide aggregations, transformations, and user-friendly final views.

Figure 1 depicts one of the main final views we created using a network of source data tables and helper-views. Other than a few source tables, such as *dbo_auction_headers_all* (information about the auction event date, time window, type, name, purchasing agent, currency, etc.) and *dbo_auction_detail* (information about line items included in the auction), all "tables" depicted in this figure are actually non-materialized views, which in turn use either source data or lower-level non-materialized views (in some cases up to three levels deep).

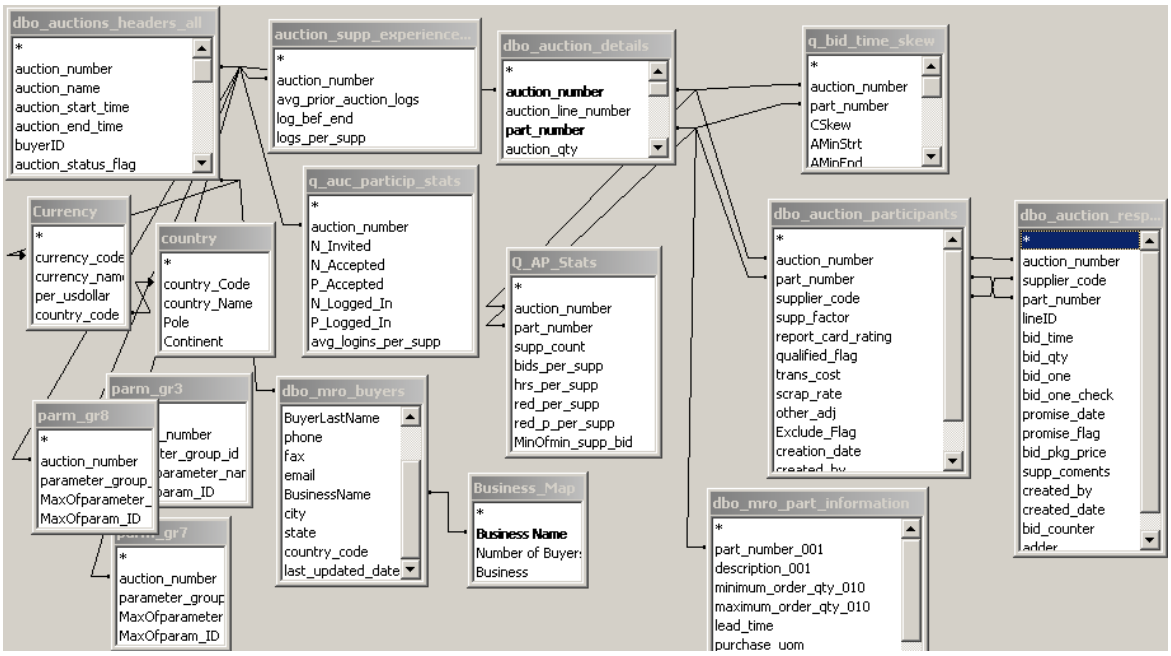

Figure 1. Source Tables and Helper-Views.

For example, the helper view *q_auc_particp_stats* provides summary statistics at the auction level (hence the foreign key link to *auction_number*) about participation levels of suppliers who were invited to participate in the auction (for example, what percent accepted the invitation to pareticipate). That view depends on two lower-level helper views that are not shown in the diagram. One of these views specializes in summarizing how many suppliers

logged into each auction. The other helper view specializes in computing for each logged in supplier, how many times they logged into prior auctions. Almost paradoxically, this complex web of helper views makes the task of addressing and adapting the transformation needs of the data warehouse much easier.

There are several benefits to this data warehouse design. First, extracting, transforming and loading new source data becomes an almost non-event. Source tables from one or several operational system are loaded into the data warehouse as simple copies, without a single change. Upon reloading of the source tables, all the views are immediately available for querying. Moody and Kortink [2000] argue for using a normalized data design for data warehouses -- we argue for retaining the original data design used in the source system.

The multi-level network of views masks and absorbs certain types of changes to the underlying source data. For example, we had cases where a table in the underlying source data was merged into another table. Adapting to this was as simple as changing one of the lower-level views to use the new data columns. Another example involved a change in how currency exchange rates were carried in the source system. Again, a change in a single lower-level view serving as the currency transformation engine solved the issue. All higher-level views, which relied on these lower-level views, remained isolated from these changes.

Ideally, each helper-view should encapsulate and specialize in the handling of a specific transformational task. This makes it easy to locate and understand specific aspects of the transformational logic. However, this design approach requires discipline in properly naming and keeping each helper view to a single independent and coherent purpose. The danger is that without such discipline, the design would deteriorate into a maze of

interdependent views. Since our design has evolved without the benefit of this hindsight, it has violated these principles in a few cases. Having a working solution, we didn't bother with a redesign of these helper views, though such rework would have been rather easy.

From the point of view of the user, this design is relatively simple. The user is exposed only to the final views, such as the one supported by the data structure in Figure 1, and hence is isolated from the underlying complexity. Such data warehouse designs are oriented towards producing very few yet very useable final views. This, in our experience, requires the involvement of a power user who understands both the reporting and data mining needs as well as the underlying transformational network of helper views.

A final view looks like and behaves as a single table. Hence, it can easily be materialized into a spreadsheet, a flat file, or a single table for data mining or reporting purposes. From the point of view of the user, the data warehouse looks like a few independent tables, each serving a particular reporting or data mining need.

4. Conclusions

Designing our data warehouse using source tables with a network of helper views leading to a few final views has helped us tremendously in conducting and adapting the ETL processes for our project. We hope that this paper prompts other practitioners to try the same approach. We also hope that future research can better identify the best approaches and design principles for designing, documenting, and deploying the network of helper views.

In the area of deployment, it is quite obvious that in many cases the underlying network of helper views and/or the final views should be materialized. Past literature has dealt with

these issues in the context of materializing aggregates views. The same approaches should yield good results for transformational views.

An intriguing idea suggested by one of the anonymous reviewers of this paper is that of developing a hybrid approach whereby the final views are arranges as a star schema. Assuming the final views are materialized (perhaps refreshed automatically upon detection of timestamp changes to the underlying source tables), this may provide a “best of both worlds” approach.

5. References

Kimball, Ralph. 1996, The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses. John Wiley & Sons: New York.

Moody, Daniel L. and Kortink, Mark A.R. 2000, "From enterprise models to dimensional models: a methodology for data warehouse and data mart design," in International Workshop on Design and Management of Data Warehouses (DMDW'2000), H. Shu M. Jeusfeld, M. Staudt, G. Vossen (Ed.). Stockholm, Sweden.

Vassiliadis, Panos. 2000, "Gulliver in the land of data warehousing: practical experiences and observations of a research," in International Workshop on Design and Management of Data Warehouses (DMDW'2000), H. Shu M. Jeusfeld, M. Staudt, G. Vossen (Ed.). Stockholm, Sweden.