Data warehousing with IBM cloud db2 warehouse

METHODS FOR DESIGN

While in the subsections above we have discussed the problems related to conceptual and logical models, in this subsection we are concerned with the techniques for building conceptual and logical schemata according to such models, considering them in the context of a comprehensive design framework that complies with good-design principles such as reusability, extendibility, and manageability.

Several techniques for automating single phases of DW design have been proposed in the literature (for instance, for conceptual design, for logical design, for physical design, for designing the ETL process). On the other hand, despite the basic role played by a wellstructured methodological framework in ensuring that the DW designed fully meets the user expectations, a very few *comprehensive* design methods have been devised so. Overall, we believe that some specific issues in design, discussed in the following subsections, have not been properly investigated yet. Besides, more generally, mechanisms should appear to coordinate all DW design phases allowing the analysis, control, and traceability of data and metadata along the project life-cycle. An interesting approach in this direction consists in applying the Model Driven Architecture in order to automate the interschema transformations from requirement analysis to implementation .

1.1 Requirements Analysis

Requirement analysis plays a key role within any software project to reduce the risk of failure. Nevertheless requirement analysis for DWs has not been given much attention so far, and it is often overlooked in DW projects mainly since (1) warehousing projects are long-term ones, in which most requirements cannot be stated from the beginning; and (2) requirements are poorly shared across organizations, unstable in time, and refer to information that must be derived from data sources .

The approaches to DW design are usually classified in two categories . *Data-driven* approaches design the DW starting from a detailed analysis of the data sources; user requirements impact on design by allowing the designer to select which chunks of data are relevant for decision making and by determining their structuring according to the multidimensional model . *Requirement-driven* approaches start from determining the information requirements of end users, and how to map these requirements onto the available data sources is investigated only *a posteriori* . Some other authors, like use some kind of mixture of these two approaches, and consider both (availability of data and user requirements) at the same time, which appears to be a promising direction of research that is superior to isolated data-driven and requirement-driven approaches. Finally, a novel approach is based on the definition of a set of *design patterns*, so that, once the needed pattern is found, it just has to be adapted to the available data and user requirements.

Though the approaches devised are promising, we believe that some further work needs to be done in order to provide designers with more usable and effective techniques for collecting information needs and quality-of-service requirements, and for translating them into (at least domainspecific, ideally general) conceptual models based on a common vocabulary between IT staff and decision makers. Thus, how quality-of-service can drive the design of the DW should be deeply studied.

1.2 Schema evolution

As several mature implementations of data warehousing systems are fully operational within medium to large contexts, the continuous evolution of the application domains is bringing to the forefront the dynamic aspects related to describing how the information stored in the DW changes over time. As concerns changes in data values, a number of approaches have been devised, and some commercial systems allow to track changes and to effectively query cubes based on different temporal scenarios . Conversely, the problem of managing *changes on the schema level* (that may be demanded by changes either in the business domain or in the user requirements or in the sources) has only partially been explored, and no dedicated commercial tools or restructuring methods are available to the designer yet.

The approaches to management of schema changes in DWs can be framed into two categories, namely *evolution* and *versioning* : while both categories support schema changes, only the latter keeps track of previous versions. If one is sure that previous schema information will never be useful again, schema evolution offers adequate functionality. Otherwise (to guarantee consistent re-execution of old reports), schema versioning offers the strictly more powerful approach. Actually, in some versioning approaches, besides "real" versions determined by changes in the application domain, also "alternative" versions to be used for what-if analysis are considered . Overall, we believe that versioning is better suited to support the complex analysis requirements of DW users as well as the DW characteristic of non-volatility. Thus, the main research challenges in this field are to provide effective versioning and data migration mechanisms, capable of supporting flexible queries that span multiple versions.

Considering the complexity of the ETL procedures, another very relevant issue is to devise techniques for propagating changes occurred in the source schemata to the ETL process. The obvious benefit in achieving these goals will be to keep the DW in sync with the business requirements, thus avoiding its obsolescence.

1.3 Quality metrics

Due to the strategic importance of DWs, it is absolutely crucial to guarantee their quality from the early stages of a project. While some relevant work on the *quality of data* has been carried out there is still no agreement on the *quality of the design process* and its impact on decision making. The most significant approaches to measuring the design quality can be framed as follows:

- *At the conceptual level*. There have been preliminary attempts towards defining metrics that allow the intuitive notions of quality of conceptual schemata to be replaced with quantitative measures (such as the number of facts, the number of degenerated dimensions, the number of shared hierarchy levels, etc.), in order to reduce subjectivity in evaluation and guide designers in their work. Obviously, the existence of a standard conceptual model could give a strong push in this direction.
- *At the logical/physical level*. Besides the recommendations and subjective criteria stated for instance in , some works were focused on quantitatively evaluating the complexity of dimensional models. Other relevant research directions include normal forms for DW and quality-driven view selection .

Overall, we believe it is necessary to devise more comprehensive metrics for measuring quality, encompassing both schema quality (to better model application requirements and to guarantee good querying performance) and data quality (to ensure timeliness of information and to take care of data aging). After their formal and empirical validation, these metrics will support the designer in evaluating and ranking different design alternatives; besides, they will be useful to better plan the project and meet user requirements, by predicting the cost and complexity of later stages in design. Particular care

should be taken in addressing the *traceability* of metrics, how metrics are translated from one phase of design to the next one, and in defining thresholds to discriminate “good” schemata from “bad” ones. Besides, techniques will be needed to *monitor* the metrics and appropriately respond to their deviations during the DW lifetime, in order to better manage extensions and evolutions. Finally, these metrics must be considered from the user point of view, by studying their impact on information analysis: methods must be devised to propagate data quality metrics to query results, like in , and to have data retrieval driven by the quality requirements expressed by users, like in .