“Data Warehousing: Lessons from Experience”

Theme

An interesting aspect of database theory is that many of the results forming the basis of the field have historically come from industry. For example, consider the papers in Stonebraker’s “Readings in Database Systems” [STON], where the majority of authors of these founding papers work in industry.

Concentration of research results in industry and away from academic institutions is not too surprising when we consider the youth of the field, and how it has been driven by commercial needs. The first databases were being created by IBM a little over 40 years ago, the first paper on the relational model was written by Codd (at IBM) in 1970, no actual relational product existed until about 1980, and we are now seeing a new object-relational standard being adopted by major vendors. In this environment, any database text written more than ten years ago is completely out of date. So where are academics to get their research insights? We are faced with a “Two Cultures” phenomenon. Database theory, in its infancy as it is, is defined in terms of the quickly evolving needs of database users, and vendors are in a much better position to track this than academic researchers. One hundred years from now, we would expect a developed theory about all the database capabilities that have proved useful, but it is still too early for this as the capabilities are changing so fast.

It is, however, important that we make the best start we can in the direction of an encompassing theory, and for that to occur, there must be better communication between industry and academia. Having worked in both environments, it always seemed to me that the two cultures have much to offer each other. In implementing a software system, a thoughtful programmer runs into many problems that deserve careful study. However, the press of events in getting a product out makes such careful study impossible, so the programmer (and even many industrial researchers) come up with short-term fixes that have no lasting merit. Now academic researchers have the time and the will to provide the careful study such problems require, and thus advance the field. What should be happening is that industry practitioners carefully note the problems they run into in practical development, and communicate these problems to their academic colleagues.

The current panel attempts to address this need, in a small way, by focusing on "Lessons from Experience" in the Data Warehousing field. The industry practitioners on the panel have all agreed to try to give an idea of the problems they believe are already well solved in the Data Warehousing field, and problems they believe need new and better solutions. The panel is meant to encourage exchanges, so after each panelist presentation we will have a short period of audience participation with questions and observations.

As an example of a potential research problem, consider the following. We know of OLAP as a form of Data Warehousing that avoids SQL, and instead takes the OLAP approach of slicing, dicing, drill down, and drill through. For large aggregations of data (100 M rows in a Sales Table), OLAP achieves much faster response time, by preaggregating all values that might be requested in some slice/dice query at any given hierarchy level within the dimensions.

It is clear that the vendors that offer OLAP solutions are taking steps to make OLAP preaggregation “indexes” a part of the database engine. Many OLAP queries can be implemented in SQL Group By or using the new Cube By operator, and query optimization practitioners, since they will have to answer such queries anyway in many cases, want to bring preaggregation under their control, by supporting it in the engine. We refer to “preaggregation indexes” because preaggregation is so similar to indexing. Both require some work in advance, take up disk space, and speed up queries of certain descriptions. The research question then, is this: How can we combine preaggregation indexes with previous indexes, in the same sense that indexes on different columns can be combined to speed up queries.

Panelists

PANEL CHAIR: Patrick O'Neil, UMass/Boston, poneil@cs.umb.edu

Clark French, Director of Software Development, Sybase IQ, cdf@sybase.com

William McKenna, Red Brick Systems, bmckenna@redbrick.com

Richard Winter, Principal, Winter Consulting, richard.winter@wintercorp.com

Dan Crowley, Principle Software Engineer, Informix, dcrowley@informix.com

References