MODELING AND ANALYSIS FOR LARGE DATABASES

David McGoveran
Alternative Technologies
13130 Highway 9, Suite 123
Boulder Creek, CA 95006
Telephone: 408/425-1859
FAX: 408/338-3113

CONTENTS

I. Introduction

With the success of relational database technology has come the ability to collect and organize large amounts of data. The standardization of data manipulation languages permits some degree of interoperability while simultaneously taking advantage of a larger work force. Although SQL was designed for ad-hoc query use and decision support, systems using it have been tuned to provide increasingly improved support for batch and online (OLTP) processing of corporate data.

In the early days of commercial relational database deployment, languages like SQL and QUEL brought tremendous new advantages to decision support and ad-hoc query applications. However, through the 1980s, the focus of relational database management shifted to production applications and corporate data. MIS managers rarely found direct access to production databases by ad-hoc and decision support application permissible. Such applications frequently produce long running and complex queries which can interfere with more time-critical processing. This fact forced the use of extracts and snapshots by ad-hoc and decision support applications.

While tools for managing larger databases have steadily improved, tools for decision support applications have been forced to adapt to production database environments, relational database transaction processing, and client/server architectures. As a result, few real advances have been made in the analytic and modeling capabilities of the technology. Decision support tool vendors have barely improved on spreadsheet technology.

The availability of large databases has led managers to expect answers to problems of ever increasing complexity. These problems are complex in several ways: the amount of data which must be examined can be in the gigabytes and may be spread over thousands of variables, and the number of inter-relationships between those variables outstrips the capabilities of the best spreadsheets.

This paper examines the key issues in performing analysis and modeling in a large database environment and takes a brief look at the available technology.

II. Large Databases

A. Definition

The definition of a large database is relative; it changes with the state of commercial database technology. When the relational model was introduced, multi-megabyte databases were all but non-existent. As late as 1986, Dr. E. F. Codd, inventor of the relational model, stated that million row tables would soon be a reality. (Actually this author had been working with commercial implementations of relational databases with million row tables for several years at that time, but the general perception in the industry was to the contrary.) More recently, databases with tens or even hundreds of gigabytes of data have become less esoteric. There are now a number of examples of terabyte databases being planned and perhaps even one or two in existence.

Large databases are motivated by many factors. The conversion of paper to electronic forms of corporate record keeping has certainly accelerated the need for large databases. Global market conditions and economic considerations have increased dependence on timely information. Acquisitions and mergers, business unit consolidation, and departmental reorganizations can all lead to integration of databases.

B. Active vs. Passive Uses of Large Databases

Beyond these trends to capture historical data, there is a variety of databases in which the data is intended to be used in a more active manner. Most of these have to do with the capture of large amounts of control information; information which can be used to determine subsequent actions. Such data is often updated and accessed many times. It may be used in mission critical applications to drive the business automatically. It may also be used in decision support applications to understand changes in the business, to find solutions to problems that did not previously exist, and to direct the subsequent course of actions. Without the proper tools, analysts resort to intuition and ease-of-access as guides in reducing the available data.

Perhaps most familiar among active databases is the idea of an enterprise database. An enterprise database integrates data from all levels of a company into a single logical database. Enterprise-wide consistency permits management to obtain reports which cross divisional and geographic boundaries. Corporate strategic planning has the potential for becoming a timely, data-driven process.

Perhaps less well-known are the demands which various applications place on database managements systems. For example, both discrete and process manufacturing systems can require the collection of massive amounts of quality control, scheduling, and production planning information. Stock and bond brokerage houses may have to maintain multiple daily quotations on tens of thousands of stocks and bonds in the process of managing investment portfolios. Most other types of financial institutions have similar problems.

Insurance companies (health, automobile, accident, life, etc.) process hundreds of thousands of claims adjustments and policy changes, requiring maintenance of records on individuals, companies, doctors, hospitals, government agencies, and so on. Telecommunications services such as telephone companies maintain information on subscribers, service options, and usage for billing and maintenance purposes. Organizations involved in medical and pharmaceutical research, particle physics, space exploration, environmental studies, and so on collect unbelievably large amounts of data from experiments.

C. Evolution of Databases

Designing and developing a large database is a non-trivial problem. When the number of entities and relationships which must be modeled exceeds a few hundred, special logical data modeling techniques and tools are needed. Detailed data flow and entity-relationship diagrams can fill walls. Their information content becomes impossible to grasp, let alone analyze for consistency of meaning, completeness, and non-redundancy. The traditional approach of integrated design becomes less feasible as the complexity of the problem increases.

While an integrated design might well have advantages, an iterative approach is often used instead. In practice, the iterative approach is usually motivated by the pre-existence of a number of smaller databases (and therefore database designs) which must be integrated. There is a variety of methods by which the data can be migrated stepwise into a new integrated database. The idea is to divide the problem into a series of pairwise integrations. During each phase of the project, conflicts in nomenclature and database designs must be resolved. This is typically followed by the elimination of redundancy where applications processing will permit. Iterative database consolidation is often fostered by such business decisions as combining profit centers, mergers, acquisitions, product line consolidation, etc.

The iterative approach may also be motivated by the need to stage an effort. Staging provides opportunities to validate and modify efforts, to spread the costs over a longer period of time, and to begin realizing a return on the investment at an earlier date than would be possible with an integrated approach. Of course, the classic motivation for iterative growth in database size and complexity is the increase in the number of applications that naturally accompanies the life of a business. New products, processes, and customers as well as new tools for management are always being developed and almost always require the management of larger amounts of data. While these can be implemented with new "islands of data", few companies perceive this as a positive approach. Contrary to the view expressed by the designer of the ANSI embedded SQL standard, database designs do not become relatively static over time – unless the company or the applications is dying.

D. Value of Data: Converting Data to Information

Although most of the concerns of corporate database administrators (DBAs) are related to managing the storage and access to data, DBAs are generally cognizant of the need to convert data into information. The costs of storing and managing large amounts of data are accepted only because of the promise that it can be used. Data is of little use if it cannot be used in a meaningful context, and this can involve a great deal of processing. The costs need to be

balanced against the potential return on investment.

The first half of the story is data collection. Update of corporate data is traditionally handled by OLTP, heads-down data entry or through batch update. Database reliability, integrity, and availability are frequently balanced against the need for performance.

In practice, the second half of the story is often less difficult to manage since it is essentially read-only. A variety of traditional tools are used to converted data into information, ranging from batch and interactive report generators, interactive query and browsing tools, and graphical query tools to more sophisticated data modeling, analysis, and processing tools.

These later tools are generally run in batch when large amounts of data or complex data relationships are involved. In order to use such tools, the user must have determined which variables are necessary to solving the problem at hand. This step can require a great deal of exploration within the database, since raw data may not be organized in a manner that makes key variables obvious.

Decision support tools may be interactive, but traditional tools of this type are generally limited to simple analysis; small numbers of variables, simple data relationships, and relatively small amounts of data. In order to use decision support tools, the key variables are usually identified by assumption and the size of the database is reduced via an extract. This process does not lead to new solutions to old problems when the business environment changes, nor does it handle new problems well.

In principle, decision support tools should be able to manage complex analysis: unlimited numbers of variables, complex data relationships, and very large amounts of data. Such tools would permit a business to make the most of its data, thereby obtaining both a higher rate-of-return and a higher return-on-investment for managing large, integrated enterprise and corporate databases.

It is extremely difficult for most data analysts to manage complex analysis without tools to help them identify key variables and relationships. Complex analysis frequently deals with qualitative non-linear relationships. Quantitative (often linear) relationships susceptible to ordinary analysis are rarely valid except over a small range of data values. When faced with a new problem, the analyst depends on heuristic knowledge (i.e., rules-of-thumb derived from experience) and testable hypotheses in order to identify those situations under which the data relationships are characterizable and predictable.

III. Modeling and Analysis

A. Some Terminology

Modeling and analysis are used for a wide variety of purposes, by analysts with diverse backgrounds, and in a variety of fields. Each field (business management, finance, econometrics, chemical research, environmental research, etc.) has its own highly specialized language. As a result, modeling and analysis tools vendors rarely adhere to a common

terminology, making it difficult to compare and contrast them. In order to discuss important features for large database modeling and analysis tools, we will take the liberty of defining a few terms.

Most people have at least a vague understanding of the notion of a variable. A variable is a way of referring to a type of data; it is used to categorize data by its meaning. The variable is often represented symbolically or by name. Through the principle of substitution, a variable may be taken to represent a particular instance of data of that type. When this is done, we say that the variable has a value.

Without identifying relationships between variables, they would not be of much use. Relationships can be mathematical functions which allow an analyst to compute the value of one variable given the values of other variables. They may also be correspondences between variables, which allow the analyst to "look up" or "locate" the value of one or more variables given the values of one or more other variables. This is similar to the way in which one uses a primary key to find a row in a relational database.

The variables which are consistently used to find the values of other variables are called parameters. They are set to parameterize the other variables. When a set of parameters can be used to access all the variables in a database, they can be said to form a parameter frame. For example, all the primary keys in a relational database form a parameter frame. Parameter frames are used to organize (or index) a set of variables.

For example, one can imagine a map of a city with a numbered and lettered grid. City Hall might be at G3, a particular bank might be at B2, and so on. The numbers and letters are parameters which can be used to look up a particular location and thus form a parameter frame. The parameter frame in this example has two special properties. First, the parameters are independent of each other; given one, the entire range of values for the other are possible. Second, every location on the map can be specified by a combination of the parameters. When a parameter frame consists of parameters which are mutually independent in this way and yet completely identify the data of interest, the parameters are called coordinates and the parameter frame is said to form a coordinate system. Because coordinates and parameters can be used to locate data, we say they give locational information.

It is useful to be able to characterize correspondences between two variables. One way of doing this is to state the number of values which will be found based on the value of a parameter. For example, if precisely one value of the variable will be found corresponding to any particular value of the parameter, we say the correspondence is one-to-one (abbreviated 1:1). If more than one value of the variable will be found corresponding to any particular value of the parameter, we say the correspondence is one-to-many (or 1:m). Correspondences can also be zero- or one-to-many, many-to-many, one or more-to-many, etc. Database designers are familiar with such relationships and often characterize entity relationship on the basis of the relative cardinalities of the entities involved.

When correspondences are not one-to-one, it may be possible to create a one-to-one relationship by aggregating a parameter or variable. For example, a database may contain information about

the salaries of employees and the departments in which they work. The correspondence between departments and salaries is one-to-many. By computing the average salary of each department's employees, we can create a one-to-one relationship between average salary and department. This process is called aggregation.

If the database also contains information about which business units contain which departments, it would be possible to compute the average salary of employees by each business unit. Similarly, the average salary of employees for all business units (i.e., for the corporation) could be computed. Because the relationships between the corporation, divisions, departments, and employees form a hierarchy of one-to-many correspondences, these computed averages belong to levels in an aggregation hierarchy. The same hierarchy might be used to compute various other quantities that refer to the level of aggregation such as sums, averages, minimums, and maximums.

If the hierarchy is well-defined, we can also perform an operation which is the reverse of aggregation. This operation is often referred to as zooming or drilling down. For example, given the average salary by division, we may wish to drill down and view the average salary by department.

It is useful to think of an aggregation hierarchy as being parameterized by level. Sometimes these levels may themselves be parameterized by a less abstract variable and given a common name or phrase. For example, we might refer to the corporation, division, department hierarchy as being parameterized by "type of business unit". A parameter like type of business unit or the more abstract hierarchy level may be thought of as controlling resolution.

When an analyst draws conclusions, the process is called data reduction. Data reduction techniques may involve computations, deduction, or inference. Regardless of technique, the process involves reducing larger amounts of data to smaller amounts of data. When meaning is associated with this smaller amount of data, the result is information.

B. Key Types and Example Uses

Analysis is a special form of modeling. Perhaps the best known form of modeling is "what-if" or predictive modeling such as that found in spreadsheet packages. What-if modeling begins with a set of known relationships among known variables along with factual data. The analyst then varies some of the base data in order to examine how the model changes the derived data. Alternatively, the analyst may also change relationships among variables or the variables themselves. This process can be used quite effectively in exploring financial questions such as changes in product pricing, scheduling and production, product line consolidation, mergers, etc.

Before an analyst can engage in what-if modeling, an appropriate set of relationships and variables must be identified within a collection of base data which accurately portrays the thing being modeled. This is the technique of empirical modeling; the process of modeling "what-is". For example, a quarterly financial report is, in principle, based on an empirical model. The relationship between actual and projected margins is a part of "what-is".

Once the analyst is satisfied that an accurate "what-is" model has been produced, attempts can be made to identify cause and effect relationships. This process of "how-is" and "why-is" modeling separates relationships between variables into causative relationships and correlations. It is known more formally as explanatory modeling. Searching for the source or cause of what is different between two quarterly financial reports requires the interplay of all types of models. "What-is" may be explored, branching off at particular moments into "how-is" and "why-is" modeling.

The analyst is often faced with raw data. In many respects the analyst is similar to a database designer. Unlike the database user, relationships of interest have not been determined. There may not be an appropriate data model which can be used to guide the modeling and analysis process.

Philosophical statements of the form "my world is fundamentally composed of [some entity such as] departments" simply serve to constrain an investigation. Though quite common, the practice is unnecessarily restrictive, failing to recognize that entities are view specific. For complex data sets, there is usually more than one way to parameterize the data. In one view, departments may be entities and products may be attributes of departments. In another view, products may be entities while departments may be an attribute of products. Department, region and product could all be used to parameterize or differentiate the data. The analyst thus has the modeling problem of determining which view is appropriate for solving a particular problem.

There are several common types of data analysis. For example, trend analysis attempts to extrapolate future values from historical data. This form of analysis is a common source of marketing and financial predictions. A particular kind of measurement is assumed to represent a simple variable (as contrasted with a compound variable) parameterized by some other variable which can be simply ordered (most often time). Some form of curve fitting is used to find a mathematical relationship between the variable and the parameter; the relationship is then used to predict values of the variable given values of the parameter.

A variation of trend analysis is used for monitoring. A common example is often found in manufacturing and quality control applications in the guise of control charts. Here, in addition to predicting values of the variable, limits on statistical variation in the values are predicted as well. If the value of the variable falls outside the permissible range, an exception report is produced and the value flagged.

Many special techniques have been developed for numerical problem determination and resolution. Problem determination is the process of defining a problem in quantifiable terms. It usually involves building a model, often one of a statistical nature. The simple variables are identified, classed as dependent or independent, relationships among them hypothesized, and these relationships classified as either causal or as statistical correlations. Once this has been done, it becomes possible to engage in problem investigation. The analyst investigates deviations between expected or desired situations as evidence by values of the variables. The values of causally related variables are then retrodicted and deviations from these values identified; such deviations are taken as evidence of the "source" of the problem providing the conclusion is statistically meaningful.

C. Pattern Identification/Interpretation

When identifying variables and relationships, the analyst must have a criterion for recognizing patterns. These techniques invariable depend on the analyst's ability to play with different views of the data. This is done by selecting different parameterizations of the data and by aggregating data in various ways. Simple graphical and charting tools can be helpful for simple analysis, but they become difficult to use for complex analysis.

It is important to be able to identify the "best" way(s) to view data. It is just as important to be able to recognize anomalous patterns, a loss of pattern identity, that data transformations have become order dependent, or that supposedly independent data transformations have become coupled. Such events are evidence that additional parameters or relationships have become important.

One common technique for pattern recognition is called cluster analysis. This algorithmic technique attempts to mimic the process by which human beings recognize visually where one object ends and another begins. Cluster analysis can be done with visual tools if the number of parameters and data points is small. Otherwise, it is usually a batch operation otherwise. The technique works well if the appropriate parameters are selected and if the number of parameters is not too large.

D. Data Organization and Aggregation

Tools which help analysts organize data can be constraining in several ways. If the tool requires that the analyst select a limited number of variables and parameters as part of the configuration, the ability to view data in new ways is limited. One the other hand, presenting an analyst with long parameter and variable selection lists from which they must choose a few is unmanageable. Multi-dimensional spreadsheets attempt to solve this problem by increasing the number of parameters and variables the analyst may select, but do not remove the limits on complexity; they neither help manage the selection process nor reduce problem complexity.

Even if a tool provides adequate support for the number of parameters and variables involved in a particular problem, it may not adequately support data aggregation. When data from various sources is to be compared, it must be compared with a common level of detail, aggregation, or resolution. The analyst may have base data and derive aggregated data from this or may be given aggregated data from which base data must be derived.

If the analyst must perform aggregation outside the tool, it is unlikely that more than a few levels of aggregation will be explored. This can seriously hamper the analysts' ability to use data from the many levels of a corporation. Possibly more important, there can be many different ways in which to aggregate data, each of which corresponds to a different view of a problem.

E. Database Issues

Tools which do not use extracts for analysis (and even some which do) attempt to access relational databases using SQL. Unfortunately, SQL is not well-designed for performing more

than the simplest aggregation tasks. Suppose you wanted to obtain the average salaries and the number of employees for employees parameterized and aggregated by department. Given an EMPLOYEE table containing EMPLOYEE#, DEPT#, SALARY, and JOB_CATEGORY the following SQL statement would do the job:

```
SELECT DEPT#, AVG( SALARY ), COUNT( EMPLOYEE# )
   FROM EMPLOYEE GROUP BY DEPT#
```

If you wanted to see the average salaries and the number of employees for employees by parameterized and aggregated by job category, you could use the following SQL statement:

```
SELECT JOB_CATEGORY, AVG( SALARY ), COUNT( EMPLOYEE# )
   FROM EMPLOYEE GROUP BY JOB_CATEGORY
```

The following statement could be used to look at the average salary and number of employees involved parameterized and aggregated by the combination of department and job category. Notice that columns referenced in the GROUP BY clause must be included in the result. This SQL requirement becomes extremely inconvenient when aggregating by many levels, even though it does insure that the result is a relational table.

```
SELECT DEPT#, JOB_CATEGORY, AVG( SALARY ), COUNT( EMPLOYEE# )
   FROM EMPLOYEE GROUP BY DEPT#, JOB_CATEGORY
```

Suppose you wanted to see this same information, but showing the count as a percentage of the employees within the department. This seems like a relatively simple requirement. Unfortunately, SQL offers no way to perform this task in a single statement! The best that can be done is to count the number of employees by department and place this information and the department number in a temporary table, then use this table to compute the average salary and the percentage of employees within the department having the salary.

```
CREATE TABLE TEMP1
   ( DEPT# NUMBER, JOB_CATEGORY CHAR( 10 ),
   AVG_SAL REAL, EMP_COUNT NUMBER )

INSERT INTO TEMP1
   ( SELECT DEPT#, JOB_CATEGORY, AVG( SALARY ),
                  COUNT( EMPLOYEE# )
   FROM EMPLOYEE GROUP BY DEPT#, JOB_CATEGORY )

SELECT DEPT#, JOB_CAT, AVG_SAL,
                  EMP_COUNT / SUM( EMP_COUNT ) EMP_PCT
            FROM TEMP1 GROUP BY DEPT#
```

If the number of levels of aggregation increases, if the user wishes to change their order, or if the number of variables being aggregated is large, the task becomes unbearably complex. It is easy to see way SQL is not usually the method of choice for analysis tasks.

Besides the complexity issue, a number of other issues arise:

- it is likely that subtle errors will creep into the required SQL
- processing costs and therefore performance are likely to suffer because many passes over the same data set may be required
- it is unlikely that any tool can automatically generate the correct SQL for any but the simplest of analysis problems
- complex SQL processing and the data results can be extremely difficult to interpret

To avoid these problems, tools frequently impose somewhat artificial limitations including:

- the number of levels of aggregation is usually limited – Even if these are not limited in the tools, most relational databases do not permit more than sixteen levels of aggregation in a single SQL statement! Yet it is not uncommon to encounter hierarchies that are hundreds of levels deep in business situations (consider a bill of materials).
- the number of variables is limited – Again, if the tool does not limit the number of variables, limitations on the ability of the relational database to handle large numbers of columns or to process more than a few tables (typically not more than sixteen) in a single SELECT statement will constrain the complexity.
- data is preorganized – Tools may force a single aggregation hierarchy or, if the user wishes to change the hierarchy, must re-scan the source database
- views of the data are predetermined – The relational model provides an opportunity for the most well-founded and solid database design available today. Unfortunately, current technology requires that the designer decide on a particular definition of an entity and model this as a table. If a different use of the data within that table is required by some application, this can lead to inconsistent application of the business integrity rules. While appropriate design choices can generally be made for a particular set of well-defined applications, the problem is unbounded when analysis problems are included in the application set. The use and definition of data in analysis and modeling is, by its nature, not rigidly defined.

If the analysis results in partial or hypothetical problem resolution and the tool has cached only a portion of the data involved in the problem, it may be necessary to identify the original source of the data. This requires that the tool maintain some form of audit trail into corporate data if subsequent analysis is to be performed.

Analysis of this sort represents a serious security issue. Authorization to perform extracts is usually managed on a non-semantic basis. These non-semantic based approaches depend on user identification or pre-packaged applications. Enforcing security in this way is not likely to match the needs of either MIS or analysts; what is needed is security enforcement based on the ways in

which the data will be used. It must be sensitive not only to the data involved and to relatively primitive SQL operations. Higher level operations as used in analysis tools and to ways of parameterizing and aggregating the data should also determine access.

Tools that depend on extracts are faced with the problem of selecting an extract strategy. Some users and tools take the "kitchen sink approach": as much data as possible is downloaded to the workstation in the hope that it might prove useful. Assuming this approach is practical in terms of redundant storage and processing time, it is a flexible. Other tools take a conservative "prove you need it" approach. These tools access the database system catalog or data dictionary (possibly copying it) and then wait for the user to select data items before performing the necessary extract. Depending on whether or not there is sufficient workstation storage, it may be necessary to maintain a working data set cache managed on a least recently used basis.

There are additional issues involved in extracting data. These include costs of formatting and reformatting the data, network transfer costs, client versus server processing (i.e., load management), and costs associated with distributed data management.

Over the last ten years, most commercial relational databases have adopted a particular technique for dealing with situations in which a data value is unknown or missing. In principle, the technique is based on three-valued logic and permits either a particular value to be assigned to a column in a particular row or for that column instance to be designated or marked as "NULL." Unfortunately, there are many problems with the standard implementation. Without going into great detail, suffice it to say that the ANSI SQL implementation is logically incomplete (i.e., wrong) and leads to database performance and interpretation problems.

For the analyst, a much richer spectrum of methods for dealing with missing data is required. In general, the analyst needs a means of assigning temporary values while maintaining the knowledge that these are "artificial." These values may be computed, assigned by simple substitution (i.e., a constant default), or assigned by some statistical distribution.

In addition to handling missing data, there is the concept of sparse data. In a relational database, it is improper to store a row in a table if the row is "empty." This makes it difficult to model situations in which it is known that the captured data is some proper subset of the whole. For example, suppose that quality control randomly selects a manufactured product to test and that a row is entered into a table for each test result along with the time, date, and source. Given only relative knowledge of production rates, both the number and the approximate production time (and date) of untested units may be inferred. In this context the test data actually collected may be sparse compared to the distribution of tests which might have been collected.

The ability to simulate uncollected tests in a controlled manner can be important, especially if the analyst wishes to model aggregations of the data. Similarly, if the test datum itself is an aggregate (e.g., an average value for power consumption of an electrical part), the analyst may wish to infer or postulate more detailed values. Few tools (and certainly no relational databases) directly support this need.

IV. Modeling and Analysis Tool Features

As noted above, modeling and analysis tools generally do not offer the flexibility and sophistication required for working with large databases. The key features fall into a few categories: presentation, support for models, parameters, variables, aggregation, and hierarchies. In this section, we list some of the features which differentiate a modeling and analysis tool for large databases from traditional tools.

- Data Visualization

The tool should make good use of visual data presentation to help the analyst identify patterns and relationship.

- Support for All Types of Models

The tool should support the creation of explanatory models, predictive models, and empirical models. While these are related, few tools are designed with that relationship in mind. This makes it difficult to move from an empirical model to an explanatory model to a predictive model.

- Explanatory Modeling

When attempting to identify the cause of a problem, two capabilities become important. First of all, data should never be compared unless it is at the same level of aggregation or resolution. Second, it should be possible to change the parameters used for aggregation freely. When a single parameter is used for aggregation, this is a simple matter. However, it is more common to have "aggregation hierarchies" which depend in several discrete parameters which are themselves ordered with respect to importance. It is even possible that no explicit parameter can be identified and that rules control the aggregation.

- Flexibility in Parameter Definition and Selection

The practical purpose of defining parameters is to be able to differentiate values and relationships; i.e. sales for department three need to be distinguished from sales for department two. The focus of the data model designer should be on deciding which ways of differentiating the data are the most useful for the purpose at hand rather than struggling with limitations on parameter definition and selection imposed by the tool vendor.

- Differentiating Between Parameters and Variables

It is important that the tool support the differentiation between parameters and variables. Aggregation, reduction, and hierarchies should each support parameters and variables, but should not confuse them.

- Identifying Parameters, Variables, and Relationships

The tool should help the analyst identify parameters and relationships. Each parameter in the system may have many levels of resolution and where users can move as easily between parameter positions, as parameter resolutions, as variable positions and as variable resolutions.

- Creating Parameter Frames

The tool should help the analyst identify parameter frames. Existing implementations of the traditional database models (hierarchical, network, and relational) are not suited for creating coordinate systems from large numbers of variables.

- Coordinate Systems

One parameter frame is often selected from other possible parameter frames because, within it, relationships between variables are "unbiased" while still differentiating data instances. Such a frame is a candidate for being a coordinate system. In a coordinate system, the parameters which make up the parameter frame need to be orthogonal or uncorrelated in order for variable relationships to be unbiased. The analyst will have to be able to look for such sets of parameter. The tool should help the analyst select such parameter frames.

- Instance Histograms

A useful way of looking for parameters that can serve as coordinates is to make a histogram of the instances of each variable. Most variables will exhibit a varying number of instances for each possible potential value. Variables which have a near constant number of instances might possibly be used as coordinates.

- Implicit Variables

It may be the case that key information (e.g., department number) was left out because it was implicit in either the structure or the meaning of the data. The tool should help the analyst make this data explicit.

- Redundant Data

It may be that certain fields in the database refer to aggregates of other fields. The tool should help the analyst identify such redundant data and either eliminate it or make the relationships explicit.

- Intuitive Time-like and Space-like Parameters

Space-like and time-like variables are frequently good candidates for inclusion in a parameter frame. Space-like and time-like variables make efficient and complete basis for many problems because (a) space and time coordinates are independent, and (b) space and time can be arbitrarily

extended as parameterizations of both resolution and location without altering the base concepts which they parameterize. As the most common, general, and frequently-used parameters currently available, the tool should make their identification and use intuitive and easy.

- Integration of Parameter Frames

The analyst may have to combine two or more different models in order to compare values and relationships within a common context. This is done by integrating the parameter frames into a new, single parameter frame. For example, a financial model of a business prior to consolidation of business units is likely to be different from the model after consolidation. In order to perform causal modeling (as when attempting to identify the cause of revenue losses) it is necessary to unify the parameter frame used pre-consolidation with the parameter frame used post-consolidation. Otherwise, spurious relationships may appear during analysis.

- Ability to Switch Parameter Frames

The ability of an analyst to model variable relations is a function of the parameter frame chosen. In particular, the ability to aggregate and analyze variables is dependent on the parameters to which they are related. If parameters are not changed methodically, values can be compared consistently.

- Groups of Variables

Many important correlations exist between groups of variables rather than between atomic variables. These can only be identified if the tool allows the analyst to group of variables easily.

- Handling Large Numbers of Variables

An analyst faced with raw data or a large database may have to manipulate hundreds of variables. It is not uncommon for tools to be limited to handling tens of variables.

- Comparisons

The tool should aid the analyst in performing comparisons between variables or parameters. When they are not aggregated at the same level, the comparison is usually ill-defined.

- Parameters and Variables

The tool should help the analyst distinguish between parameters and variables. This is particularly important with respect to aggregation; aggregating parameters can change relationships between variables and the way in which they are aggregated.

- Indicators or indexes

Another method of reducing complexity is through the creation of indicators or indexes. The Dow Jones Industrial is an example of an indicator for a complex data set. Indicators are

artificial parameters which capture an intuitive or heuristic relationship, often between an unmeasurable quality and a group of variables. Support for indicators is an important feature.

- Data Reduction

The tool should provide a variety of facilities for data reduction. The ability to perform visual pattern recognition, statistical analysis, cluster analysis, etc. can all be important. Facilities to aid deduction, induction, goal seeking, and inference can all be helpful.

- Aggregation

It should be possible to decide on a case-by-case basis whether a variable needs to be averaged, summed, or some modified combination of both. Analysis and modeling tools should not prevent the analyst from aggregating variables in complex ways. Which variables need to be aggregated through an average, a sum, or some combination of the two often needs to be determined independently for each variable. Unfortunately, tools often restrict the analyst to a single choice which is then imposed on all "dependent" variables.

- Aggregation and Parameter Frames

When selecting or creating the parameter frame, the analyst should not have to worry about how each variable will be aggregated. It should be possible to attach an aggregation function to each variable. It should be possible for the aggregation function to be a function of location and therefore parameter independent.

- Hierarchies

Traditional modeling and analysis tools do not provide methodical means to reduce problems involving very large amounts of data, large numbers of variables, or large numbers of relationships to human proportions. The ability to create and refine aggregation hierarchies consistently is necessary. Limited zooming capabilities inhibit the ability to draw conclusions. The tool should be able to manage many levels of a hierarchy simultaneously. The hierarchies need to apply to both parameters and variables. It may also be necessary to re-order a hierarchy. For example, inverting a hierarchy or part of a hierarchy can be a powerful analytic tool.

- Multiple Hierarchies and Separation of Hierarchies

For most uses, aggregation hierarchies should be made from a single ordering principle. This may require the conversion of some hierarchies into two separate hierarchies. The distinction between entities at one level of an aggregation hierarchy is usually lost by aggregating according to a different aggregation hierarchy. This situation can be remedied if the tool can be used to create and manipulate multiple parameter aggregations.

- Location and Resolution

In the same way that a parameter frame can have many locations, it can also have many

resolutions or aggregation levels. Each parameter in the parameter frame can have its own permissible set of aggregate functions. Changing resolutions should be as simple as navigating between locations.

- Missing Data

Spreadsheet tools usually require that zooming be performed on data that has already been aggregated; in this way the underlying data at a finer resolution is well-defined. Unfortunately, data is not always acquired in an unaggregated or "base data" form. As a result, it is often necessary to infer the missing details if zooming is to be performed. It is rarely appropriate to model missing data as a fixed value. More often, some statistical distribution or relationship with other base data is known. An analysis tool should provide the analyst with the ability to model such distributions or relationships.

- Documentation

An analysis and modeling tool should provide automatic documenting facilities. For example, it should be possible to keep track of which values were missing, how they were filled, and how well the gap-filling strategy seems to have worked overall. It should also be possible to keep track of data sources and the multiple methods by which derived variables may have been historically (or experimentally) computed. It should be possible to log a session, providing a trail of the operations which led to a model or an analysis.

V. Existing Technologies

Among the traditional technologies used by analysts are spreadsheets, statistical analysis packages, and expert systems. These can be divided into two categories: those which provide a general set of facilities for modeling and analysis and those which use a specific technique.

The corresponding tools based on these technologies generally do not work directly with large databases. Instead, the analyst uses an extract from the database. Depending on the difficulty of the method of extract, this can result in limiting the amount of analysis which is performed. In addition, the data is usually unaggregated and the amount of data represented is therefore limited by space available on the analyst's workstation and the limitations inherent in the tool.

Expert systems fall into the group of tools which offer a specific facility. These systems can be used to deduce conclusions given a set of rules or to induce which rules are appropriate given a conclusion. More recently a related technology has been introduced. Statistical, object oriented, and knowledge based technologies have been combined into intelligent databases. These tools work in a mode often referred to as database "mining." Unlike interactive modeling and analysis tools, such products work in batch (or on-line but as a background process) to identify patterns inherent in the data. They are not generally useful for solving specific problems, but are extremely powerful for identifying non-obvious patterns that may indicate data integrity problems and implicit business rules.

Statistical packages and spreadsheet tools fall into the category of more general tools. Most

access databases through batch extract or embedded SQL facilities. They offer a range of facilities for modeling and analysis and are usually not specific to a problem domain. A statistical package may have facilities for ANOVA (analysis of variance), Bayesian inference, factor analysis, linear and integer programming, Monte Carlo simulation, regression models, time-series analysis, and the like. Generally, they can be used for data reduction but do little to support hierarchies.

A spreadsheet is a familiar and flexible tool. It might have the ability to create sums, averages, counts, etc.. Like traditional report generators, these tools require the identification of control fields if, for example, subtotals are to be computed. Unfortunately, the problem does not necessarily suggest the appropriate control fields or parameters. The amount of work involved in changing the control fields is non-trivial.

Few products on the market today are designed to support modeling and analysis for large databases. Spreadsheets can be used to implement a model of known relationships, but not to identify the appropriate parameters or to handle large amounts of data. Expert systems and statistical analysis packages can be used for identifying relationships, but do not handle aggregation hierarchies or parameter frames.

VI. FreeThink Technology

FreeThink Technology is a new approach to modeling and analysis which attempts to overcome the weaknesses of these older technologies. Rather than being a loose collection of useful facilities, it is based on theoretical foundations. This theoretical foundation allows various facilities and techniques to be integrated into a consistent tool.

FreeThink Technology addresses most of the features described in the previous section, especially in the areas of presentation, parameters, variables, aggregation, data reduction, and hierarchies. Uniquely, its theoretical foundation allows it to treat data reduction and aggregation as strongly related operations. Its strength in these areas allow the user to perform intelligent extracts from large databases, rather than taking either the "kitchen sink" or the "prove you need it" approaches.

FreeThink Technology might be characterized as the next generation of multi-dimensional data modeling and analysis technology. In the sense that it allows users to place values and formulas in cells and connect these cells by relationships, it is similar to a spreadsheet. There the analogy ends. While this document was not meant to describe FreeThink Technology, a few key features are worth pointing out.

Unlike a spreadsheet, it gives users control over virtually any size database through its handling of aggregation hierarchies. FreeThink Technology offers visualization services for both parameters and variables. The technology provides multi-parameter zooming with unlimited zoom levels per parameter and the ability, for each parameter, to be composed of multiple dimensions.

FreeThink Technology allows the analyst to represent complex aggregation functions. For

example, representing inventory as an aggregation that sums over products and averages over months would be expressed as follows:

Inventory = avgsum[month,product]:inventory

When applied to the product group level by quarter, the inventory variable will evaluate to the average over all months of the sum, for each month, of all the products inventories.

FreeThink Technology supports the assigning of proxy functions wherever there is missing data. Thus, for example, when sales figures are missing at the department level, they can be imputed through interpolating on previous time positions. A function of this type might be expressed as follows:

Sales = Sales ?(prevtime:sales * avg sales growth rate)

In addition, FreeThink Technology offers automatic documenting facilities. It can keep track of which values were missing, how they were filled and how well the gap-filling strategy seems to have worked overall.

VII. Conclusions

Good modeling and analysis work means trying alternative solutions. Established patterns should be treated as historical facts, not behavior constraints. Unfortunately, this means that the tool must often work with large amount of data. That means that the tool must address new problems in the areas of presentation, parameters, variables, aggregation, data reduction, and hierarchies.

The existence of many variables represents the freedom to create and compare both subtle and intricate relationships. With the proper tool, the analyst is more likely to solve a complex problem, identify subtle relationships, and understand confusing situations. For today's businesses, that means better control over costs and higher profit margins.

Recommended Reading

Barry Render and Ralph M. Stair, Jr., Quantitative Analysis for Management, Allyn and Bacon, a division of Simon and Schuster, Needham Heights, Massachusetts

John O. McClain and L. Joseph Thomas, Operations Management: Production of Goods and Services, Prentice Hall, Inc., Englewood Cliffs, New Jersey

Note: This whitepaper was sponsored by FreeThink Technology, Cambridge, MA. However, all content is the considered and independent opinion of, and written under the sole control of, the author.

