illuminate

Data warehouses that don't compromise.

FLEXIBILITY SPEED CORRELATIONS

When Standard SQL Queries Can't Get the Job Done

How the value-based storage model in the Correlation DBMS provides unrestricted flexibility for data analysis and exploration.

Abstract

The technologies underlying information management infrastructures today—relational database management systems (RDBMSs) and structured query language (SQL)—are very efficient at answering standard business queries. However, they are manifestly inadequate for answering more interesting and insightful questions that can lead to innovative strategic and tactical decisions, or rich analysis of any data set.

Data warehouses contain much more valuable information that could be used to perform such analysis—if they could be accessed effectively. The fundamental shortcomings of RDBMS and SQL technology are:

> They enforce a rigid structure on the types of relationships that can be explored. Any question that lies outside the current index structure of a dataset requires an IT project to re-architect the indexing, schema or data cube.
>
> They also enforce a rigid structure on *how* questions can be asked. People can't ask the database questions in natural human language; they have to understand the relationships between columns and tables so that complex SQL queries can be written to provide the answers.
>
> They are unable to accommodate certain types of questions at all, such as incremental queries (asking a question, then asking follow-on questions based on the answer to the first query) or associative queries (finding all of the information known about a person, place, product or other element in a data set).

A radically new type of database structure—the correlation database management system (CDBMS)—is ideally structured for explorational queries that can unlock the true value contained in organizational databases and enable informed, data-driven decision-making.

By automatically and completely indexing all information while loading, the CDBMS value-based storage (VBS) structure not only significantly reduces the time-to-analytics, it eliminates the trade-off between query speed and flexibility.

Introduction

To be a serious information analysis tool, a data warehouse must respond to people in a manner that is flexible and intuitive—similar to the way people respond to each other. Without human-like interaction, data analysis systems are simply improved report writers and query systems. All intelligence has to come from the human user since there is certainly none in the systems.

Dashboards and other front-end tools currently used in many organizations can "hide" the complexity of the relational database and SQL language behind a colorful user interface. But no matter how well hidden, it is still SQL that drives the information access. And, whether there is a brilliant programmer or a creative software tool generating the SQL code, the business intelligence (BI) or other front-end analytical tool can only be used to ask questions that can be expressed with SQL.

SQL-based analytical systems can answer a question if, and only if, both a data structure and a query have been designed to answer the specific question. For example, sales by store, average margin by product and average number of products purchased by each customer are all typical queries that are available today within business BI systems. However, such systems can't answer a question like "How many of the women who are buying the new product are no longer buying a product that they had bought in the past?" unless IT spends weeks or months designing the required data structures and key relationships so a SQL language query can be written to produce the answer.

Analytical systems based on RDBMS and SQL technologies are useful for answering "typical" questions and monitoring the routine activities of a business. However, when resolving an unexpected problem or when working on a new opportunity, **the unexpected and unprepared questions are the most important ones**.

A radically new type of database architecture—the correlation database (CDBMS)—is ideally suited to answering precisely such questions. Unrestricted by the rigid structures of RDBMS and SQL technologies, the CDBMS enables people to ask difficult or unanticipated questions in a much more natural manner, and get answers in seconds. There is no need for a business user or other information analyst to "think like the database," and no need for IT to build custom data cubes or write complex SQL queries. There is no need for any trade-off or compromise between designing analytical information systems for speed versus query flexibility. And most importantly, there is no need for people to limit their questions to the "typical," backward looking, routine-activity-monitoring questions common to RDBMS and SQL information infrastructures.

The Limitations of SQL

Nearly every organization uses SQL somewhere within its information management infrastructure. Collectively, IT departments across large and small organizations have accumulated millions of hours of experience designing and developing SQL queries. SQL is built on a robust mathematical base. There are numerous academic theses defining set theory, set algebra and relational calculus. So, why is SQL a problem?

If your organization progressed with mathematical perfection in every aspect, then SQL would be the perfect tool for reporting and data analysis. But in the imperfect real world, SQL is the most important limiting factor in today's data analysis systems. It locks information into little cells that have a defined scope and purpose. No value is accessible between those cells unless someone, with very specific skills, creates a connection.

SQL in the Real World

To make these rather controversial statements a little clearer, consider some real-world business examples. Imagine that a retailer introduced a new product line and projected certain sales levels. It would take several days for an e-commerce operation or several weeks for a brick-and-mortar store to accumulate a sufficient volume of information to perform any valid analysis.

Once enough data is collected, the legacy corporate information system would tell the retailer that sales were falling short of budgeted sales levels. A multidimensional analysis system might reveal that sales of the new product were below targets in all regions and most stores. An e-commerce business may have reporting and analytical tools that measure different metrics, ranging from simple new product sales revenue to banner ad click-through rates and product page visits.

However, that is where most of today's analytical systems stop; they provide only simple reporting of facts and predefined analytics. They lack the analytical richness to enable the business to determine *why* sales are not up to expectations. Clickstream analysis may show that many prospects aren't reaching the product page, but not why those who do aren't buying. The retailer can only react to such factual information using intuition and guesses. There are many questions that an adroit analyst would *like* to ask, such as:

- What is different about the stores that are meeting the targets?
- Are the page hits we get on the new product going anywhere?
- Are there any external influences that are affecting sales?
- Who *is* buying the new product?
- Are we gaining sales in other areas from the new product promotion?
- Are we adding any new customers from this promotion?
- …

While it is theoretically possible to answer these questions using SQL, it would take an inordinately long time to create the queries. Even worse, there is no guarantee that any of the SQL queries developed would actually provide the knowledge to resolve the problem.

No one knows the right question until it has been answered.

Query Versus Question

Suppose instead that the business had an unrestricted, human-like data analysis system. A business analyst could ask the system, "What do you know about the new product line?"

This is exactly the kind of question an executive might ask a product manager when reviewing the new product performance.

Generating the required data cube structure and SQL code to translate that human question into a valid database query would take a team of experts months of calendar time. And the result would be an overwhelming volume of output that would hide the proverbial knowledge needle in a massive data haystack.

A “human” question like this requires a more natural human process to arrive at the both the initial response to the question and the desired final answer. Imagine asking this question to a person with a perfect memory, rather than a computer. That person would not simply start reciting all of his or her knowledge. They might respond with something like:

”There are 70 products in the product line, 250 stores selling those products, three vendors supplying it, 1,000 customers who have purchased and 12 who have returned one of those products. What is it that you are interested in discussing?”

In response, you might say, “I want to know more about the customers who bought one of those items. How many of them are women, and how many of those women were our customers before?”

Your assistant with the perfect memory tells you that 65% of the customers who bought the new products are women, and of those, 90% were previously our customers.

You might then ask, “Has any of those previous customers stopped buying any other products from us?” Or, you might ask, “What other products have the new customers bought?” Or, perhaps, “What about men—is the split between repeat and new customers similar?”

Obviously, there are a nearly infinite number of possible questions that could be asked in such a discovery process. A conversation like this may continue for a long time, with many questions leading to dead-ends. In a conversation with another person, a number of dead-end paths are acceptable, expected, and even beneficial since they do not consume much time and they help narrow the continuing exploration of knowledge.

Eventually, the questions lead to an understanding of the root cause of the sales shortfall. However, for a process like this to be useful, each question must be answered immediately (at worst, within a few seconds) and every possible follow-on question must be accommodated by the system.

The conversation may turn in a new direction at any time. No matter how the structure of a database is built, the need for constant redirection will frustrate every possible design. SQL may be used to create almost any defined query, but an analytical infrastructure built on RDBMS and SQL technology simply cannot answer any or all of these diverse questions in any reasonable time frame.

To have this “conversation” with a database using SQL, every possible query (a number approaching infinity in a real-world situation) would require predefined structural support (foreign keys, indices) to run properly. Additional structural adaptations (summaries, lookups, etc.) would be required to enable each of those queries to be executed with acceptable speed and provide useful results.

Once all of the required structural work was done, the queries would have to be written by an expert SQL programmer or generated by a BI system under the guidance of a very good business analyst. The quality analyst would be critical since simple execution of SQL queries in response to the human question would frequently provide inappropriate, even if correct, responses. Finally, obtaining the answers would take hours, days or even weeks, making the entire process last months or years. Clearly, for this type of sequential and natural human-like interaction with a database, SQL is simply not a viable option.

The data analysis tools in common use today are more than adequate at providing knowledge discovery, information analysis and visualization of results. They help make effective use of the mountains of data that organizations generate. However, any system based on relational storage and SQL data access is severely limited in terms of analytical power, query flexibility, and most importantly, value to the organization.

Stars, Snowflakes and Snowstorms

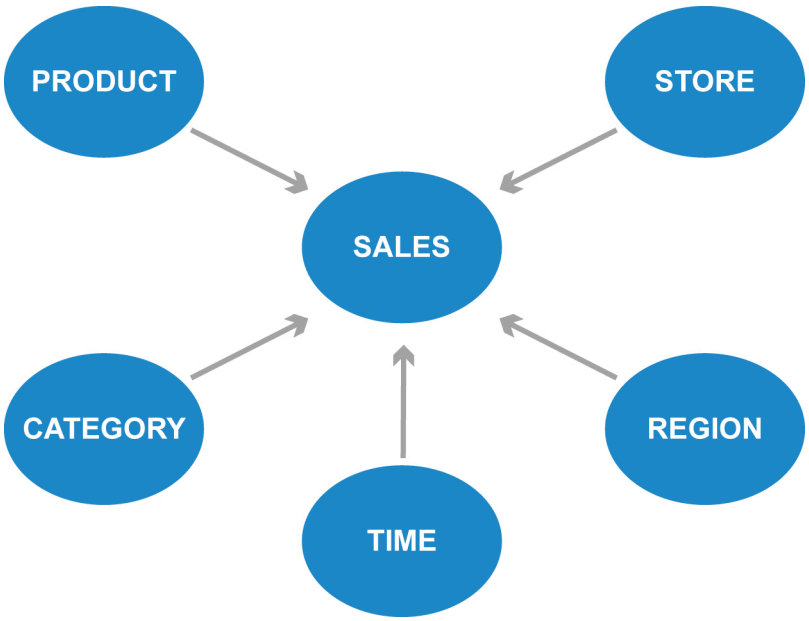
If you had access to all of the leading BI tools—such Business Objects, Microstrategy and Cognos—you could pick and choose among them to select the best possible tool for responding to each question in the business scenario above. You'd have access to the best data mining algorithms, the best multidimensional analysis reports, and the best graphing and display techniques. With the ability to apply any or all of these techniques to any and every data item stored or available from other sources, you could probably respond to any situation.

In the scenario outlined above, unrestricted data access would be crucial to accurate analysis. Assume that you are faced with this situation, but you have a well-designed data warehouse utilizing a star schema structure and a high-quality BI tool to provide analytics.

You've recognized that sales of the new product are below target and, further, that sales are low across all regions and at most stores. An easy and generally accepted response to this scenario is that the new product line is a flop and it should be discontinued.

However, some stores exceeded sales expectations. This may be a good product. But despite your well-designed data warehouse and high-quality BI toolset, due to the limitations of SQL, you realize that the time and effort required to analyze the sales results and discover the root source of the problem will be excessive and unacceptable.

The star schema shown below is a simplified yet typical representation of the structure used by many retail businesses. This schema can provide rapid analytical information relating to sales **if** the relationship that is being analyzed is one of the "points" of the star.



This structure would be useful in determining:

That sales of the new product line were low both by region and by store;

Any relationship that exists between the time of day, day of week, or other time measure and the sales of the new product

If the sales of the product category as a whole were below forecast; and

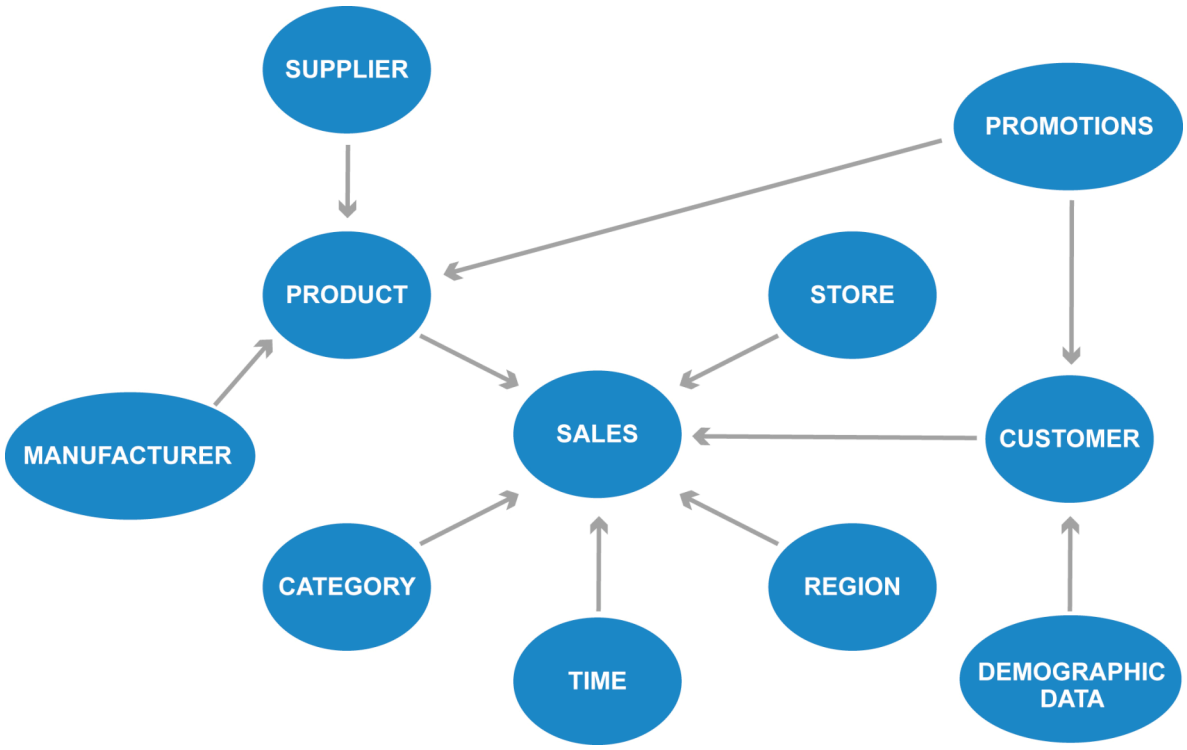
How the sales of this category related to sales in other categories.

All of these are important analyses and provide some data-driven basis for making business decisions.

However, if your analysis indicates that you need to understand more about your customers (i.e. become more customer-centric) then the star has to change. If you need better analysis of product-related information (i.e. become more product-centric), then the star will need to change again. A common approach to this problem is to develop a "snowflake" schema, wherein each point on the star may have points relating to it.

Unfortunately, in even a moderately complex real-world business, the ideal analytical schema soon looks more like a snowstorm than a snowflake.

The diagram below illustrates the result of just a little modification to support routine additional analytics.



The snowflake schema approach will quickly exceed the ability of the RDBMS and SQL query constructs to find and retrieve the data. In all but the simplest real-world applications, this model is extremely likely to fail due to the time and effort constraints imposed by SQL. In a truly complex business situation such as a merger or acquisition, the snowflake schema approach is even more unrealistic; the diagram above would look

like a spider web, with hundreds or even thousands of nodes and connections among them numbering well into the thousands.

This method is widely accepted as the best structure for analyzing complex business information—yet the associated technical restrictions make it very difficult or even impossible to utilize in most data analytics scenarios.

Life Beyond SQL

With all of these limitations, restrictions and inadequacies, why is SQL so widely used?

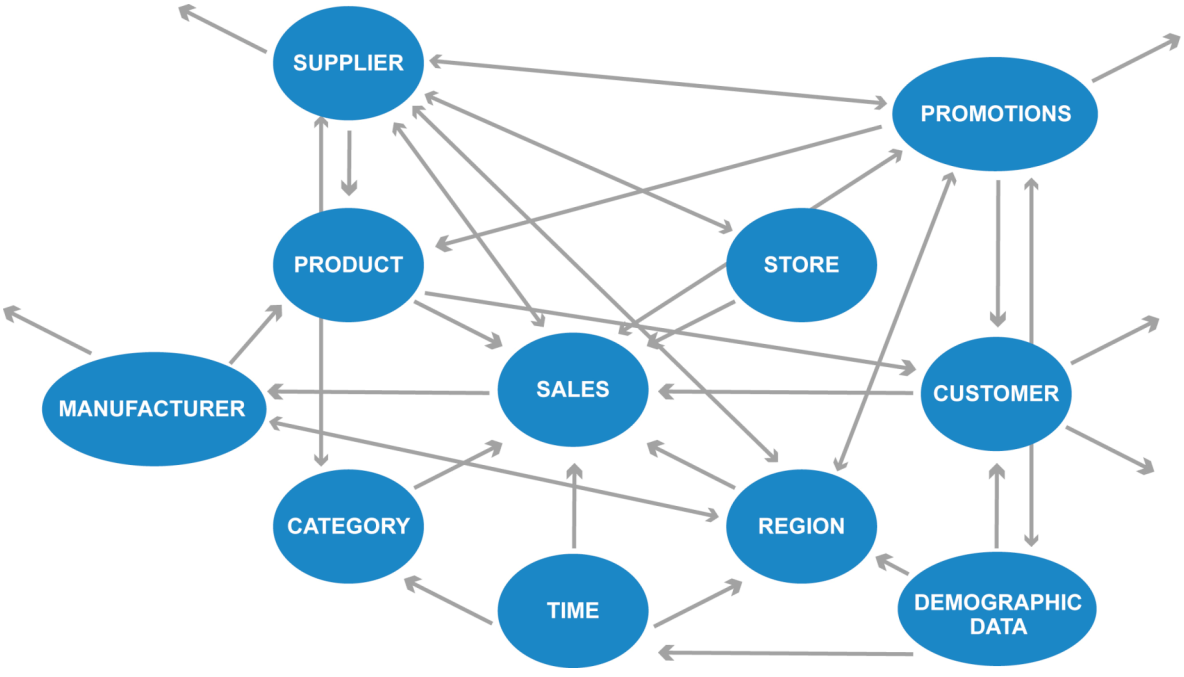
When it first arrived on the scene, SQL provided a major step forward from the then-predominant technology of hierarchical structures and procedural programming in COBOL. However, that was more than 30 years ago; essentially all of the improvements to SQL made since then have been tweaks to enhance performance or the creation of user-friendly front-end tools to hide the underlying complexity.

Any significant improvement in BI now requires a change as dramatic as of SQL that from COBOL and hierarchical structure to SQL and relational databases.

The structure needed to support such a change must have the ability to store and access any information with virtually the same flexibility as the human brain. Obviously, with a structure like this, SQL is grossly inadequate; with SQL, access to 'any' information would be restricted to questions that could be formed using set logic. There are no such restrictions on human thought or conversation.

SQL is a perfectly adequate language for concepts that can be expressed using set logic and terminology. For example, "show me the set of all sales that came from region 100" is an excellent use of SQL. However, questions such as "Where have I heard of John Smith before?" or "Why are people buying alternatives to the new product line?" do not fit into SQL constructs, yet are important questions (if, for example, you are considering loaning a large amount of money to John Smith or your new product line is failing).

To properly answer these questions, the associated data must not be restricted to any specific structure. Even a simplified "snowflake" structure would need to look like the diagram below.



Using a set oriented language to explore a structure like this would be ludicrous. The only way to find and use the intelligence contained in this data is through incremental exploration. The access method must allow a person to ask a partially formed, exploratory question and then, based on the answer, ask a series of follow-on questions until the answer is discovered. This is a "conversation with the database" similar to the dialog with the mythical assistant, described earlier.

The **iLuminate Data Warehouse,** using the value-based storage model, provides the ideal data storage structure for this type of incremental data analysis. A question like "What do you know about John Smith?" is perfectly reasonable for **iLuminate,** as all of the information in the database, in any column or table, that is associated with the data element "John Smith" can be retrieved and displayed within seconds. With this capability, an exploratory analysis process, such as analyzing the disappointing sales of a new product, can be performed at the speed of thought and produces real and immediate analytical value.

As with a relational database, the iLuminate CDBMS *can* be accessed using SQL queries. However, the iLuminate engine is not limited to SQL access, and can be used with other, more flexible access tools to perform the exploration needed to resolve unexpected or undefined business problems. Illuminate's iCorrelate exploration tool is a true ad hoc query tool, capable of fully exploiting the speed and unrestricted flexibility of the iLuminate database engine. iLuminate may also be accessed using XML, JAVA or other methods.

Asking the Database to Find "George"

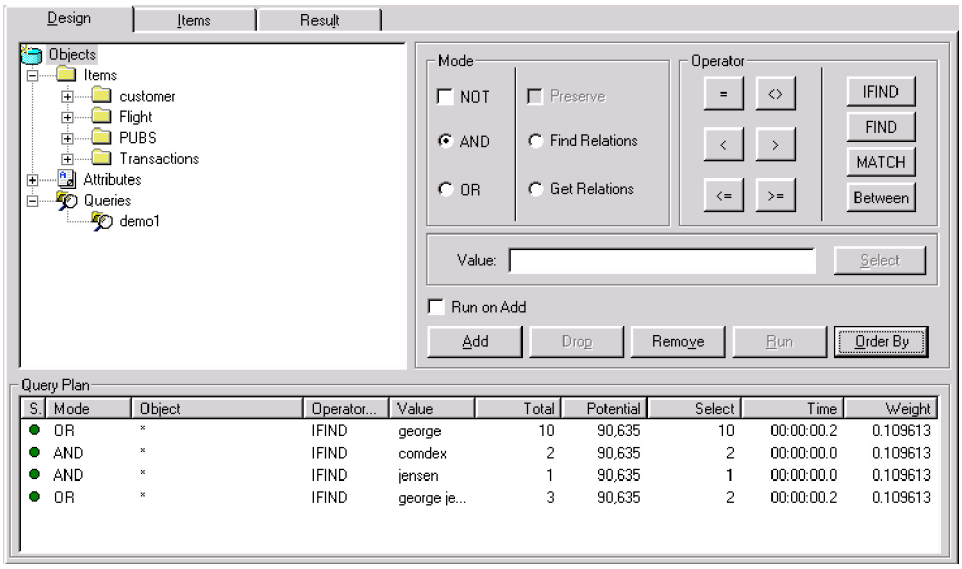
Natural language interfaces have always been challenging, due to the inconsistency of natural human language and the required consistency of computer data structures. The iCorrelate exploration tool addresses this by providing an intuitive graphical interface that enables a data analyst to ask a series of incremental questions to arrive at the desired answer—in seconds, and without any programming.

Ideally, a human user should be able to "ask" a database any kind of question that a person might be able to answer. A statement such as *"I want to call that guy I met at Comdex last year; I think his name was George. What is his telephone number, and what do we know about him?"* are reasonable questions to ask of another person, but they could never be translated into valid SQL statements, and probably couldn't be reasonably expressed in any programming language.

For these types of questions, iCorrelate's guided but unrestricted query interface is an excellent knowledge-access method. It provides a high-level view of all available sources of information and an easy way to ask questions. In short, it enables an analyst to focus on the question rather than on the method used to express it.

The iCorrelate tool answers each question quickly, enabling an analyst to continue the "conversation" one question at a time until a satisfactory answer is found. Each question can be immediately followed by new questions pursuing any line of thought.

The example on the next page illustrates how an incremental query process to find the telephone number for "George from Comdex" would develop.



S.	Mode	Object	Operator...	Value	Total	Potential	Select	Time	Weight
●	OR	*	IFIND	george	10	90,635	10	00:00:00.2	0.109613
●	AND	*	IFIND	comdex	2	90,635	2	00:00:00.0	0.109613
●	AND	*	IFIND	jensen	1	90,635	1	00:00:00.0	0.109613
●	OR	*	IFIND	george je...	3	90,635	2	00:00:00.2	0.109613

The person first asked the knowledge base to find “george.” The immediate response (0.2 seconds) indicated that there were 10 occurrences of “george” found from a possible set of 90,635 data elements. Note that the question was **not** *“Select * from customer where name equals ‘george’”* or any other structured form of a query. It was simply, “find george” with no knowledge of table names, column names or other metadata required.

The person then asked which of those 10 occurrences also contained the word “comdex.” The answer to this question (returned in 0.1 second) was that there were two records containing both “george” and “comdex.”

A quick review of the two records identified the right “George” (George Jensen) and his phone number. A third question—to determine what added information is available about George Jensen—revealed that there are three records containing information about him.

The total processing time for this interchange was less than one-half second. There were no delays waiting for computer response, or worse, waiting for an SQL query to be developed by a technician. There was no knowledge of the underlying data structure required. And there were no restrictions on the form or content of any part of the query.

Conclusion

A “conversational” query process is ideal for data analysis because it reflects normal human thought processes. The rigid structures of relational databases and SQL simply can’t accommodate such queries. In contrast, illuminate’s CDBMS structure enables people to find answers even when the question is not clearly framed; answers are arrived at through a conversational interchange with the database.

Such incremental query processes could be used to replace much of the SQL-based analysis performed today, but their real strength is in exploration. The ability to respond to incompletely formed questions, access new and undefined data, and provide interactive responses is critical for a knowledge-exploration system—for answering the interesting and insightful questions that can lead to truly innovative strategic and tactical decisions.

The restrictions of SQL or any other rigidly defined access language will never be compatible with knowledge exploration and discovery.

Data analytics systems that are limited to SQL access will never be able to provide the exploration and discovery needed for developing an understanding of a new or different business problem.

About illuminate

illuminate Solutions, headquartered in Barcelona, Spain, is the pioneer of the correlation DBMS (CDBMS) for building data-driven data warehouses. The CDBMS is a radical departure from how information management infrastructures are built and accessed. By automatically creating a data-driven schema during the raw data loading process, the need for predesign is virtually eliminated. A CDBMS data warehouse is a fraction of the size of others due to its unique value-based storage™ (VBS) model, which indexes 100 percent of the raw data on-the-fly during the loading process and stores each unique data value only once. A CDBMS eliminates the trade-off between performance and flexibility that so often frustrates IT and business users alike, and dramatically lowers the time/cost of data warehouse deployment and management.

illiminate's products include iLuminate, its CDBMS engine, and a full tool suite for accessing the data warehouse: iCorrelate, for exploration; and iAnalyze, for light analytics, dashboards and geographic mapping. The company also offers the iLuminate SDK to qualified customers free of charge.

illuminate sells its products exclusively through IT consulting services firms and management consulting companies in the Americas. In Europe, its products are sold direct and through a partner network. European customers include BBVA, Bon Preu Group, ZURICH, infojobs, Telefonica and RORI, among others.

For more information, visit www.i-lluminate.com.

About the Author

Chief architect and co-founder of illuminate Joseph Foley, is responsible for leading the company's technology vision and strategy. His expertise encompasses both the theoretical aspects of and the practical application for information structure, management and analysis. Mr. Foley co-founded Illuminate Solutions after starting and growing two successful technology companies that pioneered associative databases and other systems. With more than 30 years of experience in data management, his research and development spans industries, including retail, insurance, manufacturing, engineering, communications, transportation and more. Mr. Foley is a recognized expert in information analysis; has presented at venues around the world, including several Business Intelligence Europe forums and the DAMA/Wilshire Meta-Data Conference; and writes the Queries from Hell blog. Joe holds a BS degree in business administration with a minor in computer science.