Online Complex Processing

THE DATABASE SOLUTION FOR COMPLEX PROCESSING IN DESKTOP AND MIS ENVIRONMENTS.

DAVID MCGOVERAN

As the business community responds to today's growing global marketplace, companies find it increasingly difficult to compete. Many firms must be able to accommodate markets that change in minutes, rather than weeks or months. They must manipulate large databases in complex ways while continuing to support traditional day-to-day operations. Decision support, report generation, batch processing, and real-time data-capture applications are no longer easily separated from OLTP (online transaction processing) applications.

Most companies need some form of OLTP, along with database management systems that can support it. Nonetheless, OLTP applications are only a special, albeit historically difficult, category of applications to support with a relational database management system (RDBMS). RDBMSs designed to support OLTP generally do not address the complex requirements of today's businesses.

In addition, the popularity of client/server architectures has made it increasingly likely that businesses will use desktop-computing environments in production and even mission-critical applications. Traditional desktop applications involve a way of working that is different from that found in larger multiuser systems. It is often difficult to change the work habits of desktop users; the result is that RDBMSs in production applications must support relatively unconstrained workloads. Something besides RDBMS support for OLTP is needed, and that something has come to be called OLCP (online complex processing).

In this article I will attempt to define OLCP, describe the need for it, examine functional OLCP database requirements, and explain RDBMS features that support OLCP. First, however, a little history.

Historical Perspective

The first commercial RDBMSs were designed to support ad hoc-query, decision-support, and report-writing requirements. Those systems provided tremendous improvements over existing, prerelational products — they were easier to use, easier to manage, and it was easier to develop applications using them. As the early successes with RDBMSs accumulated, managers began to accept the new technology and encourage the adoption of RDBMSs for more diverse applications. As has happened with other new technologies, however, the early gains often led users to commit to projects without considering the potential risks.

In the early 1980s, certain application requirements were beyond the currently available RDBMS capabilities. I called them "complex database-processing" applications, and they later became known as OLCP applications. My own experience with failed relational applications helped identify and refine the specific RDBMS requirements for complex database processing — my company often performed postmortem audits and redesigned troublesome relational applications. Any of the conditions that led to one of these requirements was sufficient to place stress on available technology and products and were understood as a warning sign of impending problems. Awareness of these application requirements served as a successful guide to my company's consulting activities and our focus on difficult relational applications.

Relational applications continued to gain popularity, and more and more com-

David McGoveran is president of Alternative Technologies, a Boulder Creek, Calif. relational database consulting firm. He is coauthor of A Guide to Sybase and SQL Server *with C. J. Date, and is currently writing* An Advanced Guide to Client/Server Applications.

panies ran into problems. The main causes of these problems were that, first, the technology was still new and vendors had not (and still have not) fully implemented the relational model. Second, the difficulties of efficiently managing and using large, shared databases were not well understood. Third, most training programs did not (and still do not) focus on the proper use of relational technology, thereby creating a user community that often abused the technology.

During the mid 1980s, RDBMS vendors often heard complaints about limits on database size, the number of users who could access a database at the same time, and transaction-processing rates. The vendors misinterpreted these complaints as a need to support OLTP, rather than the broader requirements of complex database processing. Many RDBMS vendors thought OLTP support would supply all of the missing features and expand product sales to Fortune 500 firms. The term became an industry buzzword, and many vendors introduced OLTP performance features at the expense of strong relational and large-database capabilities.

What is OLCP?

InterBase, a firm now owned by Borland International, coined the phrase "online complex processing" (OLCP) in the late 1980s. The "complex" in this more marketing-oriented phrase implies significant differences between it and OLTP.

OLTP environments are relatively simple. Performance is typically measured in the 5 to 1000 transactions/second range. Most of these transactions consist of a few simple statements: Few tables are affected, few columns are updated, and updates and queries affect only one row at a time. Tables in OLTP databases usually contain only a few narrow columns, and only a few tables are either volatile or have many rows. Integrity constraints tend to be relatively straightforward and the database design is usually stable. RDBMSs used for OLTP applications require sophisticated recovery and tracking features, high availability, and support for batch processes (such as reporting and perhaps updating).

In contrast, OLCP applications must support a variety of complex operations, often simultaneously. OLCP requires high performance, but emphasizes response time and total throughput rather than transaction rates. OLCP transactions contain many complex statements that support a mixture of decision-support, interactive ad hoc-query, OLTP, batch operational processing, and batch end-user processing applications. Online decision management (OLDM) applications, which use an active database for decision support, also may be present. Because OLCP applications are usually mission critical, the high availability and performance requirements introduced by OLTP are still present; in fact, they may be greater than those in OLTP environments.

OLCP database design is sometimes unstable (time-dependent and unpredictable), and may require support for multiple versions of tables. Tables in OLCP databases are frequently volatile and contain many rows, often with many "fat" columns (columns with a relatively large amount of data per record). Integrity constraints in OLCP systems are complex.

■ The role OLTP now plays in the relational database world may well be taken over by OLCP.

The Need for OLCP RDBMSs

As users demand the promised benefits of the relational model, the role that OLTP now plays in the relational database world may well be taken over by OLCP. There are four main reasons to suggest this shift: weaknesses in available RDBMSs, mixed workload requirements, increased application complexity, and particular application types.

RDBMS inadequacies. Although RDBMS vendors have worked to improve support for OLTP applications, many have ignored the impact on other features in their products. For example, set processing is an important part of the relational model. Rather than spend R&D funds on making set-processing operations more efficient and the RDBMS "more relational," vendors focused on getting high marks in OLTP benchmarks such as TPC-A and TPC-B. Such benchmarks test the capability of systems to perform simplified, single-record OLTP operations. While there is no particular reason why an RDBMS should not perform OLTP tasks well, vendors still have far to go in developing good relational systems.

Mixed workload requirements. The often-quoted 1989 study by the Aberdeen Group of Boston pointed out that the performance needs of 90 percent of all OLTP applications could be met by an RDBMS delivering the equivalent of 12 debit/credit transactions per second. Such studies neglect the fact that these applications do not run in isolation; non-OLTP applications must be able to run simultaneously. The resulting mix of transaction profiles in an MIS environment often overloads the RDBMS — it can't handle the corresponding response-time, concurrency, and throughput tasks.

Application complexity. The complexity of applications (including performance requirements, transaction volumes, transaction complexity, and database sizes) continues to increase at a phenomenal rate, although this complexity is limited by the capabilities and the cost of the hardware and system software. Several trends suggest that the technology will not surpass the user demand:

- Users expect the now-familiar benefits of new technologies, such as text/graphic/video databases, graphical user interfaces, and cost-based query processing.
- The need for cost- and time-effective end-user training is promoting the employment of smarter user interfaces and applications that minimize human effort and involvement.
- The increasing popularity of client/server technology is introducing desktop processing habits into the MIS environment.
- Business operations are being integrated through a common logical database, with data-driven applications.
- Businesses are increasing their use of integrated design, development, deployment, and maintenance environments.
- The need for reusable software (driven by the applications backlog) further promotes data dictionary-driven and object-oriented software.
- Users want fancier number-crunching capabilities and the ability to perform them more often for up-to-date figures.

Application types. Practically every area of business has some applications that can benefit from OLCP:

- Finance: stock and bond trading and risk management (portfolio analysis and optimization)
- Manufacturing: bill-of-materials processing, dynamic scheduling and routing, and CAD/CAM
- Research: pharmaceutical, elementary particle, and military intelligence research
- Services: insurance policy maintenance, claims adjustment and reconciliation, and telecommunications provisioning

OLCP and RDBMS Functional Requirements

The presence of one or more of the following complex database-processing functional requirements may indicate the need for an OLCP application.

Fat Transactions. Transactions with many statements in them are sometimes said to be fat. The number of statements in a transaction strongly indicates the complexity of a relational application. Why? First, the number of tables an RDBMS must manage usually increases with the average number of statements in each transaction. Of course, one must be careful to make sure that record-at-a-time processing has not artificially increased the number of statements per transaction. Second, large numbers of statements in a transaction often indicate that the data manipulation language is not handling application complexity well. SQL does not provide a good method to compute descriptive statistics, for example. Third, the use of many statements suggests that the application performs some intrinsically procedural processing — processing in which the order of operations is essential.

Long transactions. Most well-designed applications do not need to support long-running transactions in which user interaction is permitted before the transaction ends. Highly interactive work is often necessary, however, in applications such as CAD. These applications often limit concurrent update requirements using operational techniques. They may require support for maintaining multiple versions of an object in the database and "nonblocking" between concurrent readers and writers.

Some computation-intensive tasks (such as risk management, manufacturing, insurance reconciliation, and so on) are best run in a batch. These have all-or-nothing transaction requirements because equivalent algorithms that would permit the application to run as a series of independent and iterative transactions may not be known or may not exist.

Complex statements. Complex statements are common in many environments: They perform mathematical computations, aggregate functions, and string functions; access multiple tables; and so forth. Statement complexity is increased by the number of columns and rows that are affected and by the number of levels of nesting.

Set processing. Set processing involves using a single statement to access multiple rows in a table. It is nonprocedural; the user does not decide the order in which the individual rows are processed. Set processing in an OLTP-based environment needs a level of efficiency seldom realized by any RDBMS. Originally, RDBMSs were designed to process these types of statements, but in an ad hoc-query environment. If an optimizer can optimize set queries, it can also optimize complex updates, inserts, and deletes, because these involve writing the result table back to the database rather than to the user process.

Highly active databases. OLCP applications often involve frequently updated data. Common examples include on-line database services, financial applications, flexible manufacturing, process control, near real-time data collection, and scientific research and development. Such highly volatile databases can stress an RDBMS.

Logical data transparency. Many OLCP applications require frequent modifications to the database schema. Each new product in a manufacturing application may require a new table, for example. It also may be necessary to redesign the database by adding new columns, or to normalize/denormalize the database for performance reasons. Similarly, stock and bond trading databases often require a new table for each new type of security. These activities place above-average importance on support of logical data transparency.

Large databases. In the 1980s, multigigabyte databases were considered large. By contrast, a particle research program might collect terabytes of data each hour. Unfortunately, no one has been able to solve the complex database processing and management problems that multiterabyte databases would introduce.

Evaluating RDBMS Support for OLCP

Each of the functional requirements discussed above places unusual demands on the capabilities of an RDBMS. Although each vendor may address these needs in a different manner, users can ascertain RDBMS support for OLCP by identifying certain key features.

An RDBMS may have difficulty handling fat transactions in a multiuser environment, especially if the RDBMS uses locks to manage concurrency. If locks are used, it becomes more important for users to have control over the granularity of locking and the degree of isolation enforced between transactions. Simultaneous support for long transactions and high levels of concurrency may preclude the use of locking techniques altogether. RDBMS support for snapshots and for multiversioning, multigenerational, or optimistic concurrency control can be extremely important as an alternative to pessimistic concurrency control and locking.

Support for complex statements and set processing demands a sophisticated RDBMS optimizer. Set processing also requires a good concurrency control scheme and scheduler. When complex statements and set processing are used in an update-intensive environment, highly volatile databases may result. Concurrency control, update efficiency, and physical resource management (such as dynamic allocation and recapture of disk space) are required to support highly volatile databases. Maintenance utilities must not compete with online users for resources. Otherwise, it might be difficult to schedule backup operations or run report applications.

The relational model traditionally emphasizes logical data transparency through views. In principle, views allow the DBA to change the database design without impacting existing applications or users. Unfortunately, not all views are automatically updatable, and few RDBMS vendors have implemented automatic view update facilities except for extremely simple views.

Alternatives to automatic view updating such as stored procedures can also insulate users from changes to the database design. Current stored-procedure technology does not modify the user's view of database tables. Instead, it is best viewed as a means to create, extend, and specialize RDBMS operations. A second alternative, triggers, allows a DBA to define update operations using stored procedures that are fired implicitly when a designated operation (such as insert, update, delete, or select) occurs against a designated table. If the designated table is a view, triggers can be used to extend the view updating capabilities of the RDBMS. The DBA must be able to anticipate what is intended by view updates.

To support very large databases, it is important that OLCP RDBMSs be designed with few intrinsic limits on the capability to create and manipulate large numbers of tables, large tables, large rows, and other elements.

With the increasing popularity of graphical user interfaces such as Microsoft Windows and Motif, users are beginning to expect text and image support from RDBMSs. Although most RDBMSs now provide some support for text and image data under the name of BLOB (binary large object) support, or even user-defined data types, few products provide integrated support for these data types. User-defined functions improve the RDBMS's handling of special data types, make applications easier to develop, and improve data integrity.

Applications involved in monitoring activities such as process control and programmed stock trading systems need to respond rapidly to changes in the database (called events). Some application structures are themselves event-driven. Traditional RDBMSs require that each application poll the database, an expensive and inefficient technique. OLCP RDBMSs provide event management, which automatically alerts interested applications when designated events occur.

By looking for these features in an RDBMS, users can obtain some degree of assurance that the RDBMS will support OLCP functional requirements. The cautious user will also examine how the features are implemented and how well they integrate with standard RDBMS features such as integrity enforcement and levels-of-transaction isolation.

Conclusions

In this article I have defined OLCP, examined its history, and discussed OLCP

WHITE PAPER *(continued)*

functional requirements and some typical RDBMS features for supporting them. This should help users recognize OLCP applications and evaluate the capability of RDBMS products to support them.

Unfortunately, RDBMS vendors have placed too much importance on OLTP. To a large extent, OLTP and OLCP represent different ways to solve business information-processing problems. The distinction between the two technologies in this regard is important. A business's need for an RDBMS that meets OLTP requirements is often driven by old, pre-relational development methodologies. Similarly, a business's need for an RDBMS that meets OLCP requirements is often driven by relational development methodologies and client/server architectures.

One way or another, RDBMS vendors must address complex database processing in the 1990s. Vendors who understand OLCP issues and give the concept more than lip service will enable users to solve many difficult application problems — and both vendors and users will benefit. ■

References and Further Reading:

• C.J. White, "What is OLTP," *InfoDB*, Spring 1989.

• D. McGoveran, "Beyond OLTP: On-Line Complex Processing," *InfoDB*, Fall 1989.

LETTERS *(continued from page 6)*

nology that, for the moment, can be satisfied with a lowly PC.

John Forester
Sunnyvale, Calif.

Sybase and SQL Server Book

My co-author and I would like to thank *DBMS* for reviewing our book, *A Guide to Sybase and SQL Server* in the September issue (page 36). However, the review contained several errors.

First, the book does not replace *Guide to SQL Server*; Aloke Nath's book remains a good introduction, focusing on the Microsoft version of the product. Second, our book is the only public source of information about Sybase's forthcoming SQL Server Release 5.0. The book was written to describe this release. Specific information about Release 5.0 is noted explicitly in many feature discussions and entire sections are devoted to it. Third, rather than discuss ANSI versus non-ANSI SQL syntax in the body of the book, the information is summarized in Appendix B (Some Differences Between Sybase and the SQL Standard). Finally, while it is true that there is no query showing a GROUP BY clause that contains fewer columns than the SELECT clause, the feature is noted in Appendix B.

The review helped identify an error in the text with regard to this topic; on page 152 the syntax is stated to be an error (the qualification "in ANSI mode" was inadvertently omitted). I would like to thank the reviewer for this and assure our readers that any such errors reported to us will be corrected in future editions.

David McGoveran
Boulder Creek, Calif.

That this book will replace Guide to SQL Server *was the reviewer's opinion, not the publisher's plan. The fact that it is the only book on the version 5.0 should help this prediction. The book is already in its second printing.*

The material in Appendix B is a list of non-ANSI features, rather than a detailed discussion of them. The reviewer would have preferred discussion in the text, but agrees that the results would have been less than easy to read. — ed.

Errata

In our *1992 Database Buyers Guide*, we failed to include VAX/VMS and DOS in the list of operating systems supported by the Ingres RDBMS in the features chart on page 27. ■