International Journal of Research in Science and Technology
(IJRST) 2014, Vol. No. 4, Issue No. I, January-March
http://www.ijrst.com
ISSN: 2249-0604

CURRENT TRENDS AND FUTURE DIRECTIONS OF DATABASE

Rohit Goyal
Assistant Professor, Himgiri Zee University, Dehradun – 248197

ABSTRACT

In present time Database Management Systems (DBMS) technology is a key technology in most Systems Integration projects. The fast pace of development in this area makes it difficult for SI professionals to keep up with the latest advances, and to appreciate the limitations of the present generation of products. This paper briefly discusses the past achievements of database research and development and comments on the current status of database technology. We then focus on new directions in this area and discuss the challenges presented by new types of applications. The existing relational DBMS (RDBMS) technology has been successfully applied to many application domains. RDBMS technology has proved to be an effective solution for data management requirements in large and small organizations, and today this technology forms a key component of most information systems. However, the advances in computer hardware, and the emergence of new application requirements such multimedia and mobile databases produced a situation where the basic underlying principles of data management need to be re-evaluated.

Keywords: *Federated, Synchronous Client/ Server Computing, Declarative Language, WAL – Write Ahead Logging.*

INTRODUCTION

Some technology observers see RDBMS technology as obsolete in the context of today's application requirements and advocate a shift towards Object-Oriented (OO) databases. The OO approach and the associated OO technologies are often seen as a universal solution in computing today, including database. While complex objects play an important role in many applications, they represent only a subset of the problems which database technology needs to address. In this paper we attempt to provide a balanced view on new trends in database technology. We discuss not only the requirement for support of complex objects (section 3.1), but also two other application areas, high-volume databases (section 3.2) and mobile databases (section 3.3), which are being successfully addressed by extending relational database technology. We firstly review the past achievements of database research and development and discuss the present status of database technology (section 2).

LATEST ACHIEVEMENTS AND CURRENT STATUS OF DATABASE TECHNOLOGY

Research and development in the area of database technology during the past decade is characterised by the striving for better support for applications beyond the traditional world, where primarily high volumes of simply structured data had to be processed efficiently. As a result, future DBMS will include more functionality, and explicitly cover more real world semantics (in various forms) that otherwise would have to be included in applications themselves. During the last decade

the early versions of RDBMS systems have evolved into mature database server technology with capability to support distributed applications and operate in heterogeneous environments. As a result of these developments, the present generation of database technology addresses the requirements of most business-style applications. Advanced database technology, however, is in a sense ambivalent. While it provides new and much-needed solutions in many important areas, these same solutions often require thorough consideration in order to avoid the introduction of new problems. One such area is database security. In this paper, we consider three prominent areas of nonstandard database technology: object-oriented, active and federated database management systems. In particular, we show which typical security problems (with the focus on access control) have to be solved for these systems. We briefly review the main achievements of database research and development in the following sections.

FROM RELATIONAL MODEL TO COMMERCIAL DBMS IMPLEMENTATIONS

Starting from the relational model formulated by E.F. Codd in 1970 and the early prototype System R developed at IBM [1] database research focused on solving practical problems associated with the management of large databases. The main accomplishments of database research in the following years include the development of high-level data management language SQL (Structured Query Language), theory of query optimization, and transaction management techniques. Numerous other technical solutions to problems related to the management of large amounts of data shared by multiple, concurrent users have been found. These include the development of buffer management, indexing, and physical storage techniques which dramatically improve the performance of commercial DBMS systems [2].

STRUCTURED QUERY LANGUAGE (SQL) LANGUAGE

SQL has undergone major revisions from the original SQL86 standard. SQL86 lacked many of the key features of the relational model including referential integrity and domains. The next version of the standard (SQL89) rectified many of the shortcomings of SQL86, and the current SQL92 standard incorporates declarative integrity, domain definitions, and numerous other features that make SQL a powerful data management language. SQL is universally accepted as the basis of all leading DBMS systems today and is likely to be the dominant database language in the foreseeable future. The current standardization efforts of the ISO (International Standards Organization) centre on extending SQL to incorporate Object-Oriented (OO) features, and provide support for complex types of data such as text and multimedia.

QUERY OPTIMIZATION

In any Relational Database Systems, Query performance is dependent not only on the database structure, but also on the methods in which the query is optimized. We show various classes of syntactically equivalent SQL queries, each of which can show off significant differences in data access depending on the features of the query formulation and the success of the database query

optimizer. Similar looking queries can take extensively different times to execute. We conclude that on-line analytical processing systems must not depend on dynamic user specified SQL queries if consistent overall system performance is required. We know that if SQL queries structured dynamically from user input, then system designers will not be capable to guarantee performance. Even if you use faster servers, this has been confirmed to be a small factor compared to the speed of the algorithm used. Therefore, the solution always lies in optimization.

SQL is a declarative language that does not specify the implementation details of database operations and uses query optimization to determine an efficient data access plan. Early versions of relational DBMSs were often criticized for poor performance. Advanced query optimization techniques based on extensive research combined with hardware advances make relational DBMSs the fastest available database technology today, suitable for operation in environments where high-level of performance is mandatory. More recently, query optimization techniques were developed for distributed databases, and effective solutions exist today for running queries across multiple databases. Further work on query optimization is likely to focus on producing techniques capable of taking full advantage of multiprocessor computer architectures.

TRANSACTION MANAGEMENT

Concurrent execution of user programs is essential for good DBMS performance. Because disk accesses are frequent, and relatively slow, it is important to keep the CPU humming by working on several user programs concurrently. A user's program may carry out many operations on the data retrieved from the database, but the DBMS is only concerned about what data is read/written from/to the database. A transaction is the DBMS's abstract view of a user program: a sequence of reads and writes. Users submit transactions, and can think of each transaction as executing by itself. Concurrency is achieved by the DBMS, which interleaves actions (reads/writes of DB objects) of various transactions. Each transaction must leave the database in a consistent state if the DB is consistent when the transaction begins. DBMS will enforce some ICs, depending on the ICs declared in CREATE TABLE statements. Beyond this, the DBMS does not really understand the semantics of the data. (e.g., it does not understand how the interest on a bank account is computed). A transaction might commit after completing all its actions, or it could abort (or be aborted by the DBMS) after executing some actions. A very important property guaranteed by the DBMS for all transactions is that they are atomic. Write-ahead logging (WAL) is used to undo the actions of aborted transactions and to restore the system to a consistent state after a crash. The two key problems associated with transaction management: concurrency and recovery have been effectively solved resulting in reliable and fast database technology capable of supporting large numbers of concurrent users. Most DBMS systems today use row-level (record-level) locking, group commits and other techniques resulting in high overall transaction rates. Similar to query optimization, transaction management techniques were extended to handle transactions spanning multiple database sites. Reliable recovery in distributed database environments is implemented using 2PC (two phase commit) protocol which maintains database consistency following failures during distributed update operations.

http://www.ijrst.com
ISSN: 2249-0604

ADVANCED DATABASE SERVER TECHNOLOGY

The emergence of client/server computing produced database server technology with advanced features designed to support operation in distributed environments. Database server systems in addition to the standard database features incorporate database stored procedures, database triggers, and remote procedure calls (RPCs). Database triggers and stored procedures improve the performance of applications, and integrity and security of data. Most database server systems incorporate facilities for data replication with support for both synchronous and asynchronous operation. Advanced database server systems are designed to operate in open systems environments and support standard APIs (Application Programming Interfaces) and gateways. For example, the ODBC (Open Database Connectivity) API developed by Microsoft and based on X/Open CLI (Call Level Interface) specification is supported by most database servers today.

NEW DATABASE TECHNOLOGY TRENDS

As noted in section 2 above commercially available database technology supports the requirements of most business-style applications. Such applications use structured data, i.e. information which can be represented as records with pre-defined standard data types (e.g. character, number, date, etc.). Business applications are characterized by well-defined (short) transactions in which individual users update simple records; typically, a single row in a database table (e.g. debit/credit banking transactions). While the management of structured data remains important other types of information including image data, audio, video, and text are used increasingly in applications today.

NEW APPLICATION REQUIREMENTS

A number of new application areas have emerged recently which require data management support that extends well beyond the traditional data management functions. Also importantly, advances in computer hardware, network and user interface technology have re-defined the context in which data management needs to perform. In this section we discuss three new database application areas: complex object databases, high-volume databases, and mobile databases.

COMPLEX OBJECT DATABASE APPLICATIONS

Application such as CAD/CAM (Computer Aided Design/Computer Aided Manufacturing), document management, and multimedia, are forcing the developers of DBMS technology to re-assess the basic underlying database principles and produce new technical solutions. The distinguishing feature of these applications is complex objects and unstructured data which are difficult to represent as tables in a relational system. Applications that use complex objects often involve complicated, long-duration transactions which can take several days or even weeks to complete. For example, transactions encountered during design activities tend to involve co-operation between several users and cannot be served effectively by traditional transaction mechanism which relies on locking and transaction rollbacks to resolve contentions. Many database researchers regard the relational model as unsuitable for applications of this type and advocate the use of object-oriented technology. OO technology has been applied successfully to programming

and user interface development, but its impact on database is still rather unclear. Several pure OODBMS (Object-Oriented DBMS) products are available commercially (e.g. Versant, O2, Object Store), but they remain focused on specialized applications and have not achieved wide market acceptance. Some research problems and many practical issues remain to be resolved before OODBMS technology can be applied to large-scale, mission-critical applications. Hybrid solutions which support both structured and unstructured information using relational and OO technologies are being developed for commercial use by database vendors. For example, Oracle Media Server [3] combines Oracle database server technology with text and multimedia servers.

HIGH-VOLUME DATABASE APPLICATIONS

Another active area of database research and development concerns applications, which use very large volumes of data. Examples of such applications include retail chain applications which store every cashier transaction in a historical database and perform ad-hoc analysis to determine customer buying patterns, and the popularity of individual products. Traditionally, the main factor limiting database performance was the speed of disk read and writes operations (I/O throughput). Advanced cashing techniques used in modern database server systems have resolved the I/O bottleneck and produced CPU-bound (central processing unit bound) systems. An obvious solution to the CPU bottleneck is to use multiple-processor systems configured as either SMP (symmetric multiprocessor) or MPP (massively parallel processor) systems. Parallel computing architectures have been used in scientific computing for some time, and they are now beginning to impact on database applications. Proprietary database machines (e.g. Terra-data, Tandem, etc.) built using specialized hardware and software were the first parallel database technologies on the market. Today, multiprocessor systems are widely available from a variety of vendors including Sequent, Encore, NCUBE, NCR, and others. The use of multiprocessor machines for database applications is becoming an attractive solution to performance problems associated with high-volume databases as the cost of multiprocessor hardware falls. MPP architectures present a particularly cost-effective solution as MPP systems are constructed using commodity processors (i.e. Intel 486 and 586) and offer excellent price-performance ratios and scalability. Several complex technical problems need to be overcome to enable database server systems to achieve good scalability on MPP architectures. To start with, MPP systems are based on shared-nothing architectures; each processor has its own RAM (Random Access Memory) and communicates with other processor via an interconnection network. Consequently conventional concurrency techniques, which rely on shared memory, do not work in this message-based environment. Also, to take full advantage of MPP architectures all database operations need to run in parallel. This applies in particular to query and update operations, but also to table and index creation, database loads and backups. Interestingly, the SQL language, because of its non-procedural nature and theoretical foundation in relational algebra, lends itself to parallelizations. SQL queries can be expressed in functional form and operations such as aggregation, sorts, joins, and table scans can be run simultaneously on multiple processors. Production versions of database server technology are available today which perform at least some database operations in parallel [4]. The impact of parallel computing on database processing is likely to be highly significant creating new opportunities for applications which cannot be accommodated with single processor architectures.

http://www.ijrst.com
ISSN: 2249-0604

MOBILE DATABASES

With the emergence of mobile computing the corporate information system also includes notebook computers and other portable devices. Similar to desktop computing users, mobile users need access to information stored on corporate database servers. In some cases, mobile users carry fragments of the corporate database with them in notebook databases and from time to time need to synchronize their information with the information held on the remote server. Mobile computing can be regarded as a special type of distributed environment and presents a number of technical problems for database implementation. Firstly, mobile computing is characterized by relatively slow communications based on wireless networks (2 - 9 kbps) or modems using phone lines (up to 28.8 kbps). Mobile communications are non-continuous and generally unreliable causing poor system availability. Synchronous client/server computing techniques which need fast and reliable networks are not suitable for communications between the server and the client application running on a mobile workstation. Asynchronous processing using store and forward techniques are best suited to this type of environment. During periods when the mobile station is disconnected, messages are stored in a queue and transmission is resumed when communications are re-established. In situations where the mobile station also carries data, the resulting environment can be characterized as a distributed database. Asynchronous replication techniques play an important role in maintaining information stored on mobile stations [5]. Database and communication technologies suitable for mobile operation were announced recently [6] and will be available commercially in the near future.

SUMMARY

New types of applications have created a situation where the existing database server technology is no longer sufficient. Both the size and the complexity of databases have dramatically increased forcing database researchers to develop new data and transaction models. The results of this research is now beginning to impact on commercially available database server technology. In this paper we have briefly reviewed the achievements of database research and development and described three new database application areas. High-volume database applications are already a reality in many organizations today and parallel database technology is beginning to be used to address the requirements of such applications. Complex object databases and mobile databases are likely to become significant areas of activity in the future.

WHAT IS THE FUTURE OF DATABASE TECHNOLOGY?

From System Integration perspective it is important to form an opinion about the future directions of database technology. Database technology selected for SI projects today needs to remain a suitable solution in the future in order to avoid expensive re-implementation of obsolete systems.

OBJECTS Vs RELATIONAL

It is now clear that object-oriented features will play an important role in future database applications. Object-oriented database techniques are necessary to accommodate complex objects and unstructured types of information. The OO approach is also highly suited for database operations in distributed environments. It is equally clear today that the OO approach is not a universal solution for all data management requirements. Relational technology is capable of further evolution as evidenced, for example, by the implementation of parallel database operations in SQL. Future database systems are most likely to include a combination of relational and object technologies covering a range of application requirements. It is likely that many of the desirable OO features will be incorporated into relational DBMS systems along the lines of the SQL3 ISO draft standard. Future versions of relational products will support user-defined data types allowing database users to define arbitrarily complex data types and corresponding methods. Support for inheritance will improve the reusability and reliability of database and application objects. The process of incorporating OO features into SQL will however take some time, and the full impact of SQL3 is unlikely to affect the user community before the end of the decade. Of more immediate practical benefit to database users will be improved reliability, security and performance of database server technology in distributed environments, and performance gains related to advances in parallel database computing.

REFERENCES

[1] Astraham. M.M. at al: System R: A Relational Approach to Database Management, ACM Transactions on Database Systems, Vol 1, No. 2, June 1976, p 97.

[2] Silberschatz, A., Stonebraker, M., Ullman, J.D.: Database Systems: Achievements and Opportunities, Sigmod Record, Vol 19, No. 4, page 6-22, December 1990

[3] Laursen, A. et al.: Oracle Media Server: Providing Consumer Based Interactive Access to Multimedia Data, SIGMOD 94, 5/94, Minneapolis, USA, page 470-477.

[4] Oracle7 Server Documentation Addendum, Part No. A120042-3, 1994.

[5] Oracle7 Symmetric Replication, Asynchronous Distributed Technology, Part No. A22542.

[6] Oracle in Motion, Technical Product Summary, Part No. A22547, September 1994.

[7] Date, C. An Introduction to Database Systems, Addison-Wesely Publishing Co., 1975

[8] Knuth, D. The Art of Computer Programming, Vol. 3, Searching and Sorting, Addison-Wesely Publishing Co., 1973

[9] *Elmasri, R. And Navathe, S. Fundamentals of Database Systems, Benjamin CummingsPublishing Co., 1989.*