IMPROVING THE PERFORMANCE, INTEGRITY, AND MANAGEABILITY OF PHYSICAL STORAGE IN DB2 DATABASES

Ram Narayanan
August 22, 2003

VERITAS ARCHITECT NETWORK

TABLE OF CONTENTS

The Database Administrator's Challenge

The physical storage environment contains the keys to three primary DBMS characteristics: speed, data integrity, and ease of administration. Determining the optimum interface between the DBMS and a storage subsystem is critical.

IBM's DB2 Universal Database (UDB) gives the database administrator (DBA) two options for managing physical data storage: native file system files and raw devices. Files defined to the file system of the DBMS operating platform offer straightforward configuration and management but have performance and integrity limitations. Raw devices resolve the performance and integrity issues but are accompanied by a significant management burden. Neither storage alternative satisfies all of the DBA's requirements and a choice must be made between speed and integrity, on the one hand, and manageability, on the other.

VERITAS™ Database Edition for DB2 gives the DBA a third disk storage alternative. Building on the VERITAS™ File System, to manage the interface between DB2 and the physical storage world, Database Edition delivers the manageability advantages of file system files, the data integrity of raw devices, and performance that surpasses both.

The File System and the Raw Device

Fast I/O response time, cast-iron data integrity, and ease of administration are cornerstones of the modern DBMS. Database processing performed by a business application commonly accounts for most of the elapsed time associated with a transaction. And, as organizations increase their reliance on information technology, the speedy response to a tap of the keyboard can mean the difference between a sale made and a sale lost, or a happy customer and a frustrated one.

Data integrity is a DBMS absolute. When a component of a business system fails, for whatever reason, it is essential that business data maintain a consistent and healthy state. A money transfer, for example, that deducted dollars from the senders account but failed before updating the recipients account would create a significant headache for a business.

As the enterprise DBMS has grown in sophistication so too has the task of administering database environments. The need to continually monitor and manage the physical storage needs of each DB2 database is a time consuming component of the DBA's day-to-day schedule.

Performance

Defining database files to a native UNIX file system gives the DBA undeniable manageability advantages. File allocation is fast and many storage management tasks can be performed without interrupting end-user access to DB2 data.

Native file system files allow the DBA to respond quickly to requests for new databases and tables, but they also exact a price in database performance. As business applications issue read and write requests for database information, the I/Os generated to retrieve and update data from the storage subsystem are processed by the DBMS engine and the native operating system (OS) file system. Buffer management, data movement, file write lock processing, and metadata maintenance are duplicated, adding unnecessary overhead to each physical I/O. Using a raw device to host DB2 data eliminates the duplicated processing of the file system.

A raw device is a pre-allocated block of space on one or more physical hard disks. The DB2 engine accesses raw device data using a character-specific device driver, rather than the mechanisms of the native OS file system. Circumventing the file system significantly streamlines I/O processing, providing the DBMS with the simplest, and most basic, method of interfacing with a physical storage device. By eliminating the redundant processing of the file system the raw device gains a clear performance advantage.

Data Integrity

Native file system files also have a potential data integrity weakness. Both the file system and the DBMS engine use system memory to speed I/O processing. Each physical I/O is cached in memory, allowing repeated reading of data from the same physical block of storage with a single disk access. Changes to cached data are periodically transferred to hard disk in a process called flushing the buffers.

DB2 flushes its buffer pool at the end of each successful application transaction. However, this action is not coordinated with the file system, which flushes its own buffer independently. The double buffering of I/O means that data DB2 believes is safely housed on physical storage may actually remain in memory. If a system failure corrupts file system cache before data is written to disk, I/Os that the DBMS engine believes have been successful can be lost. By comparison, I/O to a raw device is written immediately to physical storage eliminating the potential for data loss.

Manageability

The performance and data integrity benefits of a raw device are offset by the need for frequent, and often tedious, monitoring and management. Storage space allocated to a raw device is held captive for sole use of the database. Avoiding over-allocation, which wastes storage space, and yet sizing to allow for future growth is a careful balancing act. If the DBA under-estimates capacity, and the raw device becomes full, application access to data can be interrupted until the problem is solved. Native file system files significantly ease the task of storage capacity planning and management.

A file system file can be quickly allocated, when a database is created, and expanded, as application data grows. Additional space is allocated by the file system, with minimal intervention from the DBA. As demand for capacity grows, storage can be added to the file system, without impacting the file or DB2.

File System Files		**Raw Devices**	
Advantages	Disadvantages	Advantages	Disadvantages
Manageability • Quick allocation and automatic resizing • Add file system space online • Reduced monitoring	Performance • Duplicated I/O processing adds overhead Data Integrity • Double-buffering creates potential for data loss	Performance • Best Performance Data Integrity • Direct write to disk eliminates potential for data loss	Manageability • Continuous monitoring of capacity • Allocation requires understanding database growth needs • Adding space is complicated

Figure 1: The advantages and disadvantages of file system files and raw devices

The Best of Both Worlds: VERITAS™ Database Edition for DB2

VERITAS Database Edition for DB2 resolves the traditional physical storage trade-off between performance, data integrity, and ease of administration. Database Edition for DB2 is an integrated suite of data and storage management products, optimized for the DB2 UDB environment, and includes the

VERITAS File System. Database files allocated to the VERITAS File System provide solid data integrity, improved manageability, and, in many instances, better performance than raw devices.

The speed, integrity, and manageability benefits of Database Edition for DB2 come in large part from the Quick I/O and Cached Quick I/O features of the VERITAS File System. These components are unique to Database Edition and bridge the gap between raw devices and file system files.

The VERITAS Database Edition Quick I/O feature is implemented through a character-mode device driver and file system name space mechanism. Files flagged for Quick I/O processing are recognized by the file system and opened in raw-mode. The Quick I/O driver then uses the same methods to access database data as would be used with a raw device.

Cached Quick I/O leverages the large amounts of physical memory available, but unallocated, on many database servers. A specialized external caching mechanism, the Cached Quick I/O feature behaves like page cache, acquiring whatever virtual memory is available for use. DB2 is then able to access system memory that exists beyond the limits of the buffer pool. This is especially effective in 32-bit environments, where DB2 buffer pool allocations are restricted to 2GB.

Improved Performance

The raw-mode file access method used by Quick I/O eliminates the performance drag associated with files defined to a file system. Adding Cached Quick I/O to Quick I/O can deliver better than raw device performance.

Asynchronous I/O Support

Quick I/O gives DB2 access to the kernel asynchronous I/O feature available in many operating environments. Asynchronous I/O allows much higher I/O parallelism, considerably improving I/O throughput. However, the parallelism feature is only available when using raw device access methods.

Reduced I/O Overhead

Quick I/O reduces the processing overhead of file system I/O. The raw mode access method bypasses kernel file system buffers, eliminating the redundant copying of data between the DBMS engine and native file system cache. Instead, Quick I/O writes data directly from the DB2 buffer pool to disk storage.

The raw-mode processing of Quick I/O further streamlines the DB2 I/O process by eliminating the native file system's file write locking mechanism. File systems universally implement write locks to prevent multiple simultaneous writes to the same file. DB2 provides its own mechanism for managing I/O concurrency, however, rendering the file system's I/O serialization processing redundant.

Maximizing Sequential Pre-Fetch

The VERITAS File System is an extent-based, intent-logging file system designed for high performance and fast recovery. Disk space is allocated to files in large contiguous extents. This differs from files in a native file system, which tend to have blocks allocated on an as needed, ad-hoc, basis. Hosting DB2 data on a single file system extent maximizes the potential for sequential pre-fetch processing. When DB2 detects an application performing sequential reads against database data it begins to read ahead and pre-stage data in cache, using efficient sequential physical I/Os. If a file contains many extents pre-fetch processing is continually interrupted, nullifying the benefits.

Faster Cache

Cached Quick I/O provides the DBA with a selective buffering mechanism that increases the amount of data DB2 is able to hold in memory. Storing data in page cache, the Cached Quick I/O feature reduces the number of physical I/Os needed during read operations, giving significantly improved performance. Cached Quick I/O also uses direct-write, copy-behind techniques to improve the speed of write I/O processing.

Files suitable for Cached Quick I/O processing can be identified by monitoring cache hit ratios - any file with a high hit ratio can potentially be improved further. After enabling the file system for Cached Quick I/O, the Quick I/O and Cached Quick I/O features can be switched on, or off, on a file by file basis.

Enhanced Data Integrity

The double buffering data integrity issues, introduced by file system files, are eliminated when defining database files to the VERITAS File System. The Quick I/O driver opens database files in raw-mode, allowing DBMS write I/Os to flow directly to the storage subsystem without being staged to kernel file system buffers. A failure that corrupts the system memory will not lead to lost database data.

Greater Manageability

VERITAS Database Edition for DB2 gives the DBA the manageability advantages of files defined to the VERITAS File System. Standard system maintenance tasks, such as allocating, resizing, and defragmenting files, can be performed online, through a flexible Java-based management interface.

Richer File Management Utilities

Database Edition for DB2 uses logical volumes to provide a virtual front-end to the physical storage environment. The VERITAS File System and database files defined to the file system are allocated on logical volumes, allowing many storage management tasks to be performed online. This gives the DBA significant flexibility when managing physical storage. RAID configuration, creation and reorganization of file systems, and defragmentation can all be accomplished without impacting files or the file system. New database files can be allocated quickly, using enhanced naming conventions that make managing large numbers of files easy. The VERITAS File System files can be dynamically resized, using allocations from a pool of free space. And the familiar file management commands of the native OS can be used against file system files.

Large Database Configuration Support

VERITAS Database Edition for DB2 supports a maximum file size of 1 TB, removing the physical size limitations of many native file systems. In addition, VERITAS™ Volume Manager supports a virtually unlimited number of physical disks, or Logical Units (LUNs), contained in a disk group. Together these features eliminate both physical device constraints on file size, and the need to manage multiple file systems per database.

Conclusion

The physical storage alternatives offered by DB2 are insufficient to satisfy the DBA's need for speed, data integrity, and manageability. Raw devices, used in many performance critical application environments, have a detrimental affect on the day-to-day workload of the DBA. And file system files, which offer the DBA a means of getting out from under the time-consuming, firefighting approach to storage management, present questionable data integrity and less than optimal performance.

VERITAS Database Edition for DB2, and the Quick I/O and Cached Quick I/O features of the VERITAS File System, give the DBA a compelling third choice when designing the physical storage environment of a DB2 database. Rather than trading performance and integrity for manageability, database files defined to the VERITAS File System deliver superior performance, solid data integrity, and greater manageability.

Appendix

Benchmarking VERITAS Quick I/O and Cached Quick I/O in a DB2 AIX Environment

VERITAS™ Database Edition for DB2 undoubtedly improves the day-to-day workload of the DBA, and resolves the data integrity issues associated with file system files. The most significant advantage, however, is the performance benefit from Quick I/O and Cached Quick I/O. To illustrate the potential for improvement the benchmark below compares the relative throughput of Quick I/O and Cached Quick I/O against raw logical volumes, using the AIX Logical Volume Manager (LVM) running on raw partitions.

The benchmark test was derived from the Transaction Processing Council's TPC-C test, and is a mixture of read-only and update transactions, simulating a warehouse supplier application. The transaction mix represents the processing of an order as it is entered, paid for, checked, and delivered. This combination of processes offers a reasonable model of real-world business activity.

Parameters	System Configuration	
	RAID 1	**RAID 5**
# Users	40	120
Server	IBM p690	IBM p660
CPUs	8 x 1.3 GHz	8 x 750 MHz
Memory	8GB	8GB
Fiber Channel Interfaces	8	4
Storage	EMC Symmetrix 8530	IBM ESS F20
Disks	46 x RAID 1 LUNs (92 x 36 GB disks)	12 x RAID 5 LUNs (128 x 36 GB disks)
Interface Speed	1GHz	1GHz
Software	Database Edition™ for 1.0.1, Volume Manager 1.0.2	Database Edition™ for DB2 1.0

Figure 2: System configuration for benchmark tests

The results displayed in Figures 3 show that Quick I/O alone is able to deliver performance comparable to raw devices, and adding Cached Quick I/O results in significantly better than raw device performance.

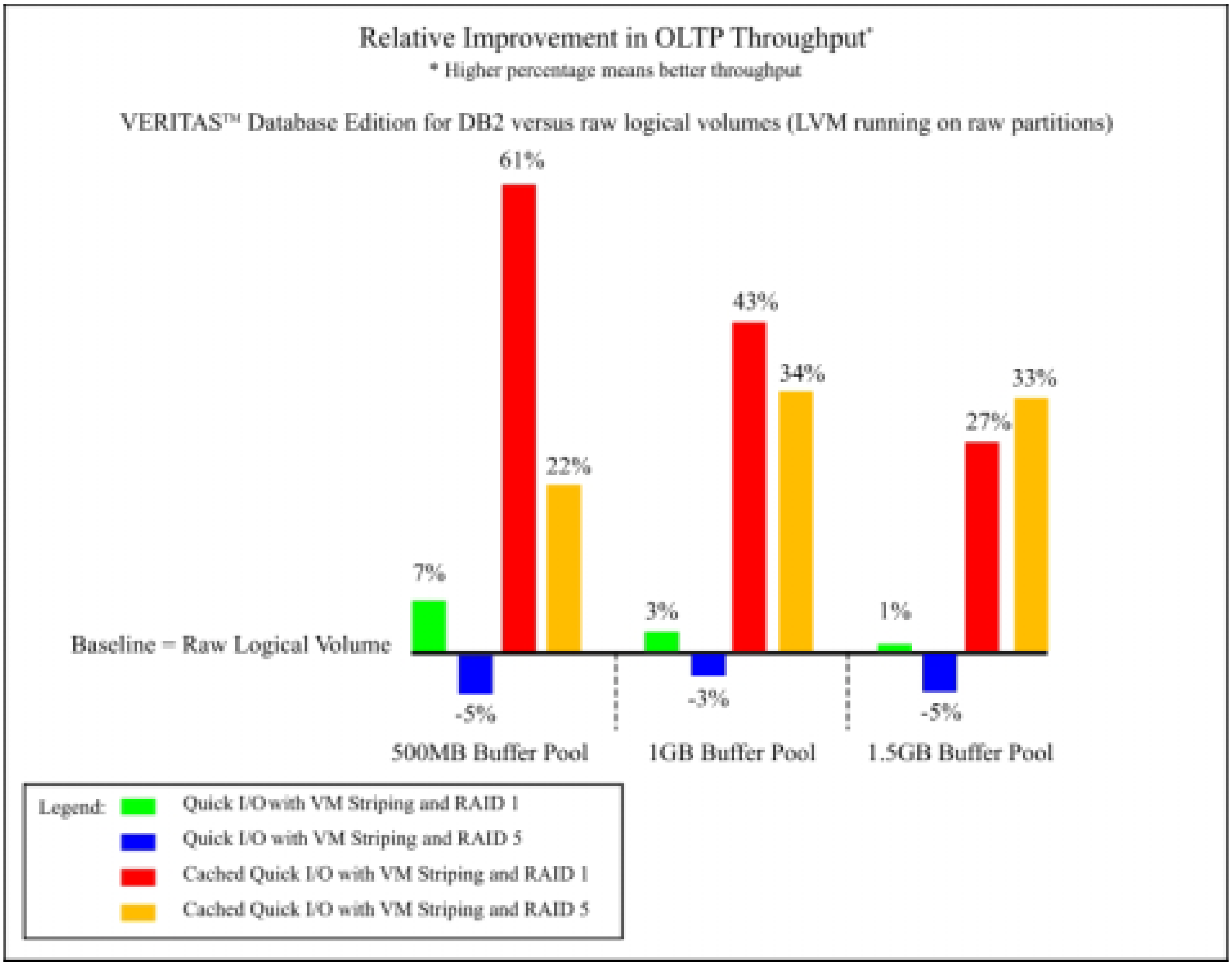

Figure 3: Comparing relative throughput of VERITAS Database Edition for DB2 and raw logical volumes

VERITAS Software Corporation

Corporate Headquarters
350 Ellis Street
Mountain View, CA 94043
650-527-8000 or 866-837-4827

For additional information about VERITAS Software, its products, VERITAS Architect Network, or the location of an office near you, please call our corporate headquarters or visit our Web site at www.veritas.com.