OS/390 allows users to measure the external input/output characteristics of DB2 page sets. An external read I/O occurs after the DB2 buffer manager was unable to locate a page within its virtual buffer pools, hiperpools or group buffer pools. External write I/Os also occur when pages that have been changed are written back to disk storage devices.

External disk I/O was once a simple function. It included basic components of disk response time: (1) access arm movement (or seek time), (2) disk rotational delay (or latency) and (3) transference of data across a channel. If the overall workload placed on a physical disk volume or the path from the central processor through data channels and disk control units was too great, delays due to these bottlenecks would occur.

The newest disk control units use sophisticated microcoded logic and increasingly inexpensive and massive storage for caching data. Caching data has two major benefits. If a page can be found in the cache when performing an external read, the slow operations of seek and latency are avoided. The page is transferred into central storage at channel speed. For example, a 4K page requires less than one millisecond on the newest equipment. Likewise on an external write to disk, the freshly modified page can be moved into the control unit cache at channel speed. Logic in the controller is then used to perform the physical write to the target volume without causing a DB2 subtask to wait for seek and latency.

This presentation is designed to help DB2 system and database administrators understand external cache and disk performance statistics. By sharing this information with their OS/390 performance and storage administration colleagues, a program to improve DB2 I/O performance can be implemented. A methodology is discussed whereby external I/O performance characteristics can be used to (1) intelligently select options such as DB2 buffering, (2) categorize page sets by relative importance and (3) optimize the use of cache storage in the disk control units.

DB2 system administrators and database administrators as well as disk storage administrators and OS/390 performance analysts all have unique skills and experiences. Working together, they can form a synergy to improve DB2 performance while reducing the I/O subsystem workload and increasing OS/390 throughput. This joint effort is hampered by the specialized training, vocabulary and experiences of the groups. All talk about “hit ratios.” The DB2 specialists will be referring to the success of look-aside buffering within DB2 while retrieving pages of interest in any of DB2’s internally managed data caches (the DB2 virtual buffer pools, hiperpools and group buffer pools). The OS/390 specialists will be referring to the success ratio of locating pages within the cache storage of external disk control units.

Therefore, the first goal of this presentation is to encourage communication between these people. The presentation includes a review of the terminology used by each and to give precise, unambiguous definitions. There is a review of the three data caches offered by DB2 and the fourth cache option using the modern disk storage control units.

This presentation provides an analytical methodology to facilitate communication between the specialists. This analysis requires the accurate I/O statistics at the page set level that are available from SMF, as well as supporting information from DB2 statistics and accounting reports. Three actual measurements from installations are offered to illustrate the analysis and the resulting conclusions and recommendations that improved both DB2 buffering and control unit caching.
Disk has been the major bottleneck in obtaining throughput and fast response when working with large data stores. The sum of seek time, rotational latency and data transfer for a 4K-physical block has not improved that much since the disks offered with the IBM System/360 of 1964. The 2314 disk on those early systems accessed a 4K-page in an average of 86 milliseconds. The newest IBM Ultrastar 2XP disks used in the RAMAC Virtual Arrays take 12.1 ms.

That is only an 86% improvement. Meanwhile, processors have progressed much farther. Comparing the IBM System 360/65 with the fifth generation CMOS (complementary metallic oxide semi-conductor) processors shows that speeds have increased 190 times. Using as many as ten processors in a single OS/390 image takes that ratio to 1,900 to 1. A fully configured 32 member parallel sysplex offers nearly a 36,000% increase! Clearly, the slower disk I/O system needs careful attention for good performance with on-line transaction processing (OLTP), ad hoc query, on-line analytical processing (OLAP) or batch.

In like manner, economics have changed dramatically for central storage (core, memory or RAM in the vocabulary of others). Not too many years ago a processor with one megabyte of storage was considered large. We are now passing through a period when two gigabytes or more is termed large. Central storage is the ideal data caching medium. DB2 exploits this with larger virtual buffer pools and hiperpools. Data sharing in a parallel sysplex configuration adds even more cache in coupling facility group buffer pools.

The people who best understand the implications of minimizing external I/O through the use of DB2 buffering are database administrators. By focusing on external I/O, DBAs should be able to more effectively balance the use of virtual buffer pools, hiperpools and group buffer pools versus disk control unit cache. The goal is to match page set characteristics with the buffering and caching configuration. The insight gained from observing both internal I/O with DB2 measurements and external I/O statistics by way of SMF 42-6 records provides the basis for more intelligent decision making.

Direct access storage devices such as magnetic drums and disks have been available for over forty years. Optical disks and compact disks (CDs) arrived later. The vast majority of direct access storage devices, typically termed “DASD,” are magnetic disks. As this paper only discusses DB2’s use of direct access storage, the term disk is used in place of the more general DASD.

The early disks gave birth to on-line transaction processing (OLTP) and saw the coming of transaction processing management systems such as IMS and CICS. Sequential files on magnetic tape have no capability to compete with randomly accessible data on disk storage units. On-line access drove the need for database management systems such as IMS’S DL/I and DB2 that provided an easy-to-use application programmer interface, concurrent access control mechanisms plus logging and utilities for recovery/restart. However, random access is slow access. Tape is read sequentially, packing many logical records into one large physical block, and is very efficient if the vast majority of the records accessed had to be processed during one run. Random access to a single record on a tape is extremely slow and, thus, not feasible.

The cost of random access was easy to measure. When the request from the application reached the access method, the application went into wait state while:

1. The disk access arm moved to the appropriate cylinder of the disk--seek.
2. The disk surface where the record was recorded rotated under the read/write head--the rotational delay or latency.
3. The data was read from or written onto the disk surface.

Disk access is measured in milliseconds. In human terms, a few milliseconds isn’t much. In computer terms, milliseconds are like hours.
The flow of logic in performing an external I/O involves four major elements.

OS/390 Data Facility Product (DFP) initializes a channel program containing both channel command words (CCWs) and disk control unit orders. This channel program is passed to the Input/Output Supervisor (IOS) of the operating system. The request is queued on a unit control block (UCB) representing the logical disk volume. If the UCB is handling multiple requests for DB2 and other users, the request is subject to a queuing delay. If multiple requests are already queued for the UCB, DB2’s request goes to the end of the queue using first-in, first-out logic (FIFO). Priority I/O scheduling (supported starting with DB2 V3) is available for placing DB2 requests higher in the queue. But if queue delays are noticed in a DB2 production system where performance is essential, the UCB is overloaded with work. Priority I/O scheduling will help DB2 have faster access to the UCB but off-loading some of the work or spreading it out over more UCBs are better long-term solutions. Generating more UCBs (after all, they are only logical devices) is the preferred solution. Every third generation disk control unit has a maximum number of logical devices that it will support. Be certain to specify that maximum to obtain the maximum number of UCBs.

After any queuing delays at the UCB level, a pending delay (PEND) may be incurred if all paths to the device(s) or the access arm(s) is (are) busy. Multiple access arms are used with various implementations of RAID (Redundant Array of Inexpensive Disk) technology offering vastly improved reliability of data storage on disk devices. Increased performance is also claimed but actually RAID only offers a faster data transfer rate across the channel—seek and rotational delays are still heavy contributors to random access time. True performance improvements are offered by external caching opportunities in the disk control units. Seek and rotational delays are tallied under the heading disconnect time. The fourth elements of external I/O are track searching and data transfer contributing to connect time. The total disk read operation is the sum of queuing delay, pending delay, connect time and disconnect time.

The Second Era

Small Cache Control Units
< 64 MB of Cache Storage
Some (KB) Non-Volatile Storage

Goal:
• Faster Batch
  Sequential Read and Write
Using:
• Read-Ahead
  Track Caching
• Write-Behind
  Block Caching

Basic Microcode

The original disk control units contained a minimum of logic and no storage for data caching. The controllers responded to “control unit orders” included within channel programs developed in the mainframe and passed from the mainframe to the channel to the controller. As storage and logic economics improved, disk entered its second generation with the original microcoded control units with what are now termed small amounts of cache storage—typically less than around 64 megabytes.

Some of the cache storage was equipped with electrical battery back-up in the event of a power failure. This is called non-volatile storage (NVS) as it can retain its contents for some amount of time using battery power. Non-volatile storage is extremely useful for writing data to disks. The data can be moved from central storage into the controller’s cache and the mainframe program does not have to wait for the seek, rotational delay and write. This removes a large delay in any writing function and, with the battery back-up, avoids loss of updated information when ordinary electrical power fails.
On the reading side, on-line transaction processing wasn’t the consideration--sequential batch processing was.

Sequential runs best when sufficient cache is available to read ahead, or to prestage data, and to write-behind, or destage. The application’s main task is “helped” by two subtasks per data set with all three (or more) tasks running asynchronously--one reading, one processing and one writing. Cashed disk controllers permit an extension to this logic. When data is to be read from the disk and the seek and rotational latency are complete, this type of disk controller not only reads the requested physical block, it also reads the remainder of the blocks on that track and places them in cache storage. When the next read request came, the data was transferred from cache without the seek and latency delays.

The performance of destaging updated data back to disk was greatly improved with non-volatile storage. One or more physical blocks of updated data are transferred from central storage to controller cache. The central processor is signaled that the operation has been successfully completed when the channel has completed the data transfer. If the application logic was waiting for the I/O complete signal, it could now be redispached by the operating system. Meanwhile, the disk controller would “spool” or destage the data from its cache back to the disk surface.

These two functions were so successful that later models of controllers read more than one track at a time and some even read the rest of the cylinder. This is excellent for batch sequential processing but of no value to random on-line work. In fact, In-line systems, it is undesirable to tie-up the control unit and volume for the duration of lengthy prestaging operations.

Another problem that needed to be addressed was the dynamic nature of data sets. Over the course of a day’s processing schedule, data sets could display quite different characteristics. The characteristic of primary importance was the total number of I/O requests over a given time period. Many data sets are uniformly accessed over time--some at a consistently high level while others are consistently low. Consistent data sets are much easier to diagnose and derive an optimum caching scheme.

The more difficult data sets are those that show widely varying I/O requests over the course of the processing day.

Further complicating the issue is the mode of access. Typically, during prime shift for online processing, a data set may be 100% randomly processed while during off-hour batch and utility processing, the access mode is 100% sequential.

The manufacturers of disk control units attacked this problem with more sophisticated microcode within the controllers.
This second generation saw a great number of advancements as the cost of logic and storage continued to improve. The basic movement was more storage for more read caching and more non-volatile storage for better destaging writes.

The microcode was also continually being upgraded although often in concert with the access methods and I/O supervisor of MVS and now OS/390. The two goals were a better balance between sequential and random plus the ability to dynamically react to changing patterns of access to a data set.

The first goal was realized by enabling the mainframe to signal the disk controller to bypass prestaging data as the read request was coming from a random process not a sequential. Rather than calling this capability “prestaging avoidance,” the more positive terms “record caching” or “bypass caching” have been adopted. This has been extended from RC I to RC II (record caching levels 1 and 2). A heavily random process can now signal the disk controller that the data set being used is so random and lightly accessed that it doesn’t pay to save the record in the controller’s cache. When read from disk, the data is moved to the mainframe’s central storage and not retained in the controller’s as is normally done.

The second goal is realized by more sophisticated logic that tracks individual data set performance over time and adjusts the use of the disk controller’s cache accordingly. The two stages of evolution of this logic are called Dynamic Cache Management and Dynamic Cache Management Extended (DCME). DCME works like a data set thermostat. When the data set is lightly or randomly accessed, prestaging is not used and record caching may be bypassed. As the data set “heats up” with higher access intensity, DCME tracks random versus sequential and can turn on record caching and, eventually, read prestaging.

RAID (Redundant Array of Inexpensive Disks) technology offers new levels of physical data integrity but with the loss of the identity of physical disk volumes. The new controllers spread the logical data over multiple physical disks to support RAID recovery schemes. All RAMAC virtual arrays and EMC² systems have very large cache storage (very large today--nothing special in future years) exploiting the low cost of CMOS storage. They house increasingly sophisticated dynamically adaptable microcode to off-load disk access management responsibility from the mainframe. One EMC² representative joked that if they put a transaction manager in their controller microcode, you wouldn’t need a mainframe. More correctly, the controller would have become a mainframe.

But this anecdote illustrates the power that now is available in what was once a simple control unit. The current control units also illustrate how far we’ve come to minimize the negative effects of the slow seek and rotational latency on application performance. It also shows a “fuzzy line” between who is really “in charge” of optimizing I/O performance--the DB2 buffer manager and its caches or the microcode logic and cache in the disk controller.

The balancing of internal and external caching is really the heart of this paper because both are useful even though they are the responsibility of two different groups in an installation--the DB2 people versus the disk storage specialists. It is hoped that this information will help minimize the “versus.” Both groups and the installation benefit from sharing their knowledge and formulating the best configuration of data caches.
### DB2 4K Page Read/Write Times

<table>
<thead>
<tr>
<th>Queuing Pending Disconnect (Seek &amp; Rotation) Connect (Search &amp; Transfer)</th>
<th>3380E</th>
<th>3380K</th>
<th>3380J</th>
<th>3390-3</th>
<th>3390-2</th>
<th>3390-1</th>
<th>U 2XP</th>
</tr>
</thead>
<tbody>
<tr>
<td>25.3</td>
<td>24.3</td>
<td>20.3</td>
<td>22.1</td>
<td>19.6</td>
<td>16.6</td>
<td>11.7</td>
<td></td>
</tr>
<tr>
<td>1.4</td>
<td>1.4</td>
<td>1.4</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>26.7</td>
<td>25.7</td>
<td>21.7</td>
<td>23.1</td>
<td>20.6</td>
<td>17.6</td>
<td>12.1</td>
<td></td>
</tr>
<tr>
<td>- - -</td>
<td>- - -</td>
<td>- - -</td>
<td>20%</td>
<td>20%</td>
<td>20%</td>
<td>50%</td>
<td></td>
</tr>
<tr>
<td>Weighted Average</td>
<td>26.7</td>
<td>25.7</td>
<td>21.7</td>
<td>18.6</td>
<td>16.7</td>
<td>14.3</td>
<td>6.1</td>
</tr>
</tbody>
</table>

To quantify the three generations of disk control units, this chart shows the access speeds of common IBM disk storage units over the sixteen year life of DB2. In all cases the queuing and pending time are shown as question marks. They should be nearly zero in a well configured, balanced system. Therefore, minimizing queuing and pending time is one goal of external I/O tuning.

Connect time has improved by 50% based mostly on the faster ESCON data channels now available. Disconnect time, mostly the seek operation, has also be almost cut in half. This shows a bit of the frustration with disks. They are electro-mechanical devices and physically moving the access arms can only be improved so much. Therefore, the greatest gain is actually derived by caching that has ranged from zero in the non-cached, first era controllers to 50% or higher in the latest generation. Without the cache hits, disk progress has really not improved to match the rate of improvement of the central processors.

Therefore, there is the common DB2 adage, “The best external I/O is no external I/O.” Exploit the three highest levels of DB2 caching as much as possible to avoid external I/O. When external I/O cannot be avoided, DB2 can benefit from second and third generation disk controllers’ cache storage for reading. There is a definite benefit from non-volatile storage for writing. What is needed is an approach to optimize all four levels of caching for DB2 read performance improvements.

### Caching Efficiency

| A. Random GETPAGE Calls | 193,239 |
| VBP Hits | 171,141 (A-B-C) 88.5% |
| VBP Misses | 22,098 (B+C) |
| Disk Cache Controller Hits ??? |

| B. Hiperpool Hits | 14,054 (7.3%) |
| Disk Reads ??? |

| C. Synchronous Reads | 8,044 (4.2%) |

Here again is a possible point of conflict between the specialists in DB2 and I/O subsystems.

The DB2 tuner, looking at their System Statistics report can evaluate three basic measurements:

A. The number of random GETPAGE calls issued by the DB2 data manager while retrieving rows on behalf of using threads.

B. The number of pages found by look-aside logic searching the hiperpool.

C. The number of synchronous reads required to satisfy the random GETPAGE request.

From these three measurements, the virtual buffer pool hits and misses can be calculated as shown by the equations. Overall, the virtual buffer pool and hiperpool are delivering a 95.8% internal hit ratio. Conclusion: internal caching is saving a great deal of synchronous reading. The allocation of buffering space is good enough.

Conversely, the disk I/O subsystem specialist can look at their statistics and see that 8,044 external read requests were issued. For two extreme examples, assume first that there are zero hits in the disk cache and then compare that with, say, 8,000 disk cache hits. In the first case, disk cache isn’t providing any benefit to DB2 and increasing the size of the controller cache is pointless. In the second case, the disk cache hit ratio is 99.5% and, at first glance, looks great. But, if the allocation of disk cache can deliver a high random hit ratio, wouldn’t increasing the size of the DB2 buffer pool display even better performance at no great difference in marginal cost? Would this not also greatly reduced external I/O? Excellent disk cache hit ratios while servicing random DB2 requests are a signal to increase the DB2 internal cache.
DB2 offers an installation parameter, SEQCACH. The default value is BYPASS with an optional SEQ value. The purpose of this parameter is to give the installation control of how DB2 interacts with the disk I/O subsystem cache storage when doing read operations. Note that this option applies to all DB2 page sets (data sets) used by one subsystem.

Originally, DB2 architects thought that the performance of sequential prefetch was good enough to negate the use of disk controller cache for sequential operations. Obviously, DB2 would prefer no caching on its random reads from disk. Therefore, BYPASS is preferred with second generation disk controllers with their limited amount of cache storage.

The SEQ option is intended for those with third generation controllers where much larger amounts of disk cache storage are available. This option causes DB2 to signal that disk caching is to be used with DB2’s sequential operations.

Conversely, the microcoded logic of the disk control unit is dynamically applied to each page set as a separate object. With the second generation controllers, experience with Dynamic Cache Management Extended (DCME) has resulted in very sophisticated logic in the third generation. A simple illustration which may or may not apply to specific controllers in an installation (depending on engineering change levels and microcode releases), shows that the disk caching characteristics vary from one time period to the next based on the performance during the preceding time period (usually measured in minutes). Below a certain percentage of cache read hits, say 10%, caching is completely turned off for the page set. That is, blocks are transferred from disk to central storage but are not retained in disk cache. Between 10 and 55% cache read hits, track caching may be used while, above the 55% threshold, cylinder caching is turned on.

The basic checklist of recommendations for providing the best possible service for DB2 I/O requests has been produced by IBM specialists in their Santa Teresa Laboratory.

From the OS/390 standpoint the best options are to install the latest RAMAC virtual array technology with as much control unit cache storage as possible. Each configuration of RAMAC or similar storage arrays can support some maximum number of logical disk volumes. These relate to operating system unit control blocks (UCBs). The OS/390 system generation should include one UCB for each of the maximum number of logical volumes. This minimizes the potential for queuing delays on a UCB.

Overall, a disk cache hit ratio of 50 to 80% is a basic target to shoot for. In the case of DB2 page set, hit ratios at the high end of this range would suggest that more DB2 buffering space would be an attractive alternative.

Because the new arrays use the concept of virtual disks, don’t spend a great deal of time on volume contention problems. If you have experienced a great deal of work in this area with first and second generation controllers, you should appreciate the fact that the new microcoded logic is continually working to balance the workload on its physical volumes within the virtual logical volume framework.

DB2 options include (1) specifying the SEQCACH SEQ option, (2) utilizing as much DB2 buffering space as practical and (3) taking advantage of DB2 compression as well as the hardware compression offered by the third generation controllers. Anything to reduce the bulk of the data stored on disk increases the effective transmission rate of data across the channel interface. Finally, with third generation controllers, the SnapShot approach to making backup copies of DB2 objects is a major performance enhancement.
Methodology to Tune External I/O Performance

1. Collect SMF 42-6 and DB2 Statistics and Accounting Over Long Time Periods
2. Format Report with Suggested Calculations
3. Sort SMF 42-6 by Highest Access Intensity Page Sets
4. Analyze:
   a. High Cache Usage — DB2 Buffer Pool
   b. Control Unit Microcode Candidates
   c. Logical Volume Contention Conditions
   d. DB2 Directory (DSNDB01), Catalog (DSNDB06) and Sort Work File (DSNDB07) Characteristics
   e. Class 3 I/O Wait Time Within DB2 Class 2 Elapsed Time
5. Storage Administrator, OS/390 Performance Analyst and DBAs Review Reports and Make Changes Based on Business Objectives
   a. Storage Administrator - Disk/Cache Resources, DFSMS Storage Classes
   b. OS/390 Performance Analyst - Central and Expanded Storage Resources
   c. DBA - DB2 Buffer Pool Configurations, Application Page Sets and Critical Plans and Packages

A team methodology for improving external I/O performance is proposed. This methodology is based primarily on the OS/390’s System Management Facility (SMF) 42-6 records. These records should be collected over a minimum of a full prime shift of work. A maximum can be a full week. A good historical base is key for evaluating changes. DB2 Statistics and Accounting reports should also be generated for these time periods.

There may be thousands of DB2 page sets so it is important to have an effective way to identify those with the highest impact. The key metric is the total load on the page set and is derived by multiplying the average disk service (or response) time in seconds by the I/O rate. This gives the “access intensity.” The access intensities for the page sets are sorted into descending sequence so that the most critical are listed first. Typically, the 80-20 rule does not apply to DB2 page sets. Often only 20 to 80 page sets account for the lion’s share of external disk I/O.

After identifying critical page sets, their characteristics are analyzed to identify candidates for the application of higher performing alternatives. Basically, if a page set is experiencing high controller cache hit percentages, there is a high probability it would benefit from more space in a DB2 pool. This increased buffering can also be achieved by isolating the critical page sets in a dedicated tailored pool and by segregating “low” hit percentage page sets into different pools. Page sets with low and medium cache hit percentages may be candidates for DCME management or Record Cache technology. DB2 prefetch activity may benefit from controller sequential mode access by taking advantage of the control unit’s multi-track prestige capability. Contention between multiple critical DB2 page sets on the same disk volume can be identified by high queue time. Finally, identification of the DB2 directory, catalog and temporary work file external I/O can provide insights into when adjustments are needed for those system page sets.

Because the storage administrator, OS/390 performance analyst and DB2 DBA all manage different resources, it is important that they review reports together. Tuning recommendations should be based on business objectives and have full management support.

DFSMS OS/390 SMF 42 Subtype 6 Major Fields

- SMF Header
- Product Section
- JOB Header
  - Job Name (...DBM1, MSTR)
  - Performance Group
- Data Set Header (One per active data set)
  - Data Set Name
  - Data Set Type (LDS)
- Data Set Information
  - Average Response Time
  - Average Connect Time
  - Average Disconnect Time
  - Average Pending Time
  - Average Number of I/Os
  - Number of 3990-Extended Platform Compatible
  - Number of Cacheable I/Os
  - Number of Read Cache Hits
  - Number of Write Cacheable I/Os
  - Number of Write Cache Hits
  - Number of Record Cache Requests
  - Number of DCME Inhibit Misses

Check the Layout for Your Current Release of OS/390

The key fields of the SMF 42-6 record are identified on this chart. The records are written when a page set is closed or an SMF interval expires. Capturing the 42-6 records requires two things: (1) SMF interval recording must be enabled for started tasks such as DB2, CICS and IMS; and (2) SMF 42 records must be specified for recording and collection. Interval recording is most useful for long running jobs such as the DB2 Database Services Address Space (DBM1).

SMF 42-6 records are only written for page sets with at least one disk I/O during the interval. With DB2, multiple page sets are included in one set of interval records. Remember, however, if the DB2 buffer manager satisfies a GETPAGE request with a virtual buffer pool, hiperpool or group buffer pool hit, external I/O information is not incremented. The SMF 42-6 record will show the net difference between the number of GETPAGE calls issued to a page set and the sum of hits in the virtual buffer pool, hiperpool and group buffer pool. Small, very active DB2 page sets may show almost no external I/O because of good buffering. The SMF 42-6 records will, however, include all page set write operations. The key information deals with disk service times: the averages for queuing, connect, pending and disconnect times.

This information is recorded regardless of whether the page set is DFSMS managed. It is recorded and written for most OS/390 access methods. All of the access methods used by DB2 are included. The job name recorded is the DB2 subsystem identification concatenated with “DBM1” for the Database Services Address Space or “MSTR” for the System Services Address Space. All DB2 application I/Os are performed by “DBM1”–all DB2 log I/Os are under “MSTR.”

Additional cache read/write statistics are available for disk storage controllers with cache. These include cache read write hit counts and the number of sequential operations involving prestaging. Use of Record Cache requests and DCME activities are made available if supported by the control unit.
**DB2 SMF 100 and 101 Usage**

- **SMF 100 - DB2 System Statistics**
  - Buffer Pool Sizes and Thresholds
  - Counters Used to Calculate Virtual Buffer Pool and Hiperpool Hit Percentages (Random, Sequential, List Prefetch)
  - Count of Times Critical Buffer Pool Thresholds Were Reached (e.g. Number of times sequential prefetch disabled)

- **SMF 101 - DB2 Application Accounting**
  - Application Plan/Package Usage Statistics
  - Values to Calculate Where Time was Spent During Application Execution

To complement the SMF 42-6 information, optional SMF 100 records provide DB2 system services statistics written periodically as specified in the initialization parameter STATIME. Thirty minutes is a recommended interval. Most counters in the SMF 100 records are accumulated since DB2 was last started, so the totals include counts prior to the reporting period. Among many other uses, SMF 100 produces statistics for each virtual buffer pool and hiperpool including their defined sizes, threshold values and whether any of these critical thresholds have been exceeded. Buffer pool hit ratios can be calculated for random, sequential and list prefetch logical GETPAGE calls. These ratios are useful for historical tracking as they are key indicators of the success or failure of adjustments made to buffer pool parameters.

SMF 101 records provide DB2 application accounting data. If requested, they are written when a thread terminates or is reused with a new authorization identification (Auth ID). Accounting reports can be sorted into descending sequence by total elapsed time to identify applications that use the majority of resources in the system. The 80-20 principle applies--20% of the application programs use 80% of the resources. SMF 101 records identify resource-intensive programs with the most potential for improvement in their use of SQL and access paths.

This is the proposed layout for reporting the contents of the SMF 42-6 records. Some of the fields are taken directly from the records while others are calculated. It is strongly recommended that the report be sorted in descending sequence by the page set access intensity. The column definitions are provided below. Many report generating products capable of producing this layout are available.

**VOLSER:** If the page set is multi-volume, only one disk volume serial number is usually reported. APAR OW20866/PTF UW34547 provides improvements for multiple volumes.

**Access Intensity** is the disk response time (Disk RESP) multiplied by the I/O rate and measures how busy the volume is servicing the page set. Totals for a page set or physical volume higher than 0.3 indicate overloading in an on-line transaction processing environment. Queuing will occur for the disk volume, delaying disk response time therefore increasing application elapsed time. The page sets such a volume should be analyzed. Those with high I/O rates are candidates for DB2 buffering or disk control unit caching. Note that both VOL001 and VOL002 are above 0.3 for their access intensity and show queue time (QUE) of over 60 milliseconds each. These disk volumes are both heavily overloaded.

**I/O Rate** is equal to the total number of I/Os to the volume divided by the time interval of the report (usually measured in I/Os per second).

**Disk RESP** is the average elapsed service (or response) time for performing external I/O.

**QUE** is the time the I/O request waited in a queue because of a busy path or access arm. QUE is the pending (PEND) time, connect time (CON) and disconnect time (DISC) subtracted from the total service time (RESP). QUE is completely wasted time caused by short- or long-term overloading of a physical access path.

---

**SMF 46-2 Report**

<table>
<thead>
<tr>
<th>Data Set ID</th>
<th>VOLSER</th>
<th>Access Intensity</th>
<th>I/O Rate</th>
<th>Disk RESP</th>
<th>QUE</th>
<th>CON</th>
<th>DISC</th>
<th>RW</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS01</td>
<td>VOL001</td>
<td>0.853</td>
<td>14.05</td>
<td>60.7</td>
<td>37.0</td>
<td>2.9</td>
<td>20.0</td>
<td>2.8</td>
</tr>
<tr>
<td>DS02</td>
<td>VOL002</td>
<td>0.585</td>
<td>9.63</td>
<td>60.8</td>
<td>36.1</td>
<td>3.5</td>
<td>20.3</td>
<td>0.7</td>
</tr>
<tr>
<td>DS03</td>
<td>VOL003</td>
<td>0.054</td>
<td>4.63</td>
<td>11.7</td>
<td>2.3</td>
<td>1.6</td>
<td>7.3</td>
<td>149.6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data Set ID</th>
<th>VOLSER</th>
<th>OA Hit %</th>
<th>CHIT %</th>
<th>Read HIT %</th>
<th>DFW HIT %</th>
<th>DCME INHI %</th>
<th>RC</th>
<th>SEQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS01</td>
<td>4.6</td>
<td>59.1</td>
<td>45.3</td>
<td>97.5</td>
<td>91.9</td>
<td>33.1</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>DS02</td>
<td>10.4</td>
<td>75.0</td>
<td>41.8</td>
<td>98.9</td>
<td>85.1</td>
<td>27.8</td>
<td>3.0</td>
<td></td>
</tr>
<tr>
<td>DS03</td>
<td>67.6</td>
<td>75.1</td>
<td>74.9</td>
<td>95.3</td>
<td>9.8</td>
<td>3.6</td>
<td>1.0</td>
<td></td>
</tr>
</tbody>
</table>
Report - Column Definitions

| Data Set ID | DB2 Page Set Identification |
| VOLSER     | Disk Volume Serial Number   |
| Access Intensity | I/O Rate multiplied by Disk RESP |
| I/O Rate   | I/O Rate in Channel Programs per Second |
| Disk RESP | Total Response (Access) Time (in ms.) |
| QUE        | Queue Time Waiting for the UCB |
| CON        | Channel Connect Time (Data Xfer & Protocol) |
| DISC       | Channel Disconnect Time (Seek and Rotation) |
| R/W        | Ratio of Disk Read to Write Operations |
| OA Hit %   | Overall Disk Cache Hit Percentage |
| CHIT %     | Cacheable Disk Hit Percentage |
| Read HIT%  | Disk Cache Read Hit Percentage |
| DFW HIT %  | Disk Cache Fast Write Hit Percentage |
| DCME INHI %| Percentage of Time DCME Inhibits Caching |
| RC %       | Record Cache Read Request Percentage |
| SEQ %      | Disk Sequential Request Percentage |

CON is the time data is being transferred between the disk and channel. For a 4-K page, this should be one to two milliseconds. If there are longer connect times, it may be prefetch I/O (more than one 4K-page per channel program) or the use of an 8, 16 or 32K-page size.

DISC is the time it takes for the access arm to move to the correct cylinder on the disk (seek time) and for the required disk record to rotate to the position of the access arm’s read/write heads (rotational delay or latency). The channel is disconnected from the control unit and disk during this period and is free to perform other work. Disconnect time also includes rotational positioning sensing (RPS) misses plus staging and interface busy delays within the controller.

R/W is the read-to-write ratio and is only valid for cacheable I/O. Because SEQCACH BYPASS (used by DB2) and INHIBIT (used by DCME) I/O are not cacheable, this activity is not included. Sequential reads are not counted as cacheable I/O. A very high R/W ratio with a high read hit percentage, indicate a page set that will benefit from a larger virtual buffer pool/hiperpoo combination. The other extreme, a low read-to-write ratio, may indicate that DB2 is externalizing modified pages to disk without needing to read the data. This indicates a high rate of update over a small number of pages. The pages remain in a buffer pool for reuse but updates are frequently flushed to disk (see DS02).

Hit% is the overall hit percentage within the controller’s cache storage and is the total of cache hits divided by all of the I/O requests (cacheable and non-cacheable).

CHIT% is the cacheable hit percentage (total cache hits divided by cacheable I/Os). When (1) the CHIT% is equal or slightly higher than the HIT%, (2) the CHIT% is less than 40% and (3) the DCME% is 0, the page set is a good candidate to use DCME.

Read Hit% is the hit percentage for cacheable reads.

Decision Table for Cache Hit Categories

<table>
<thead>
<tr>
<th>High Cache Read Hit Percentage (&gt;50%)</th>
<th>Disk &amp; Disk Cache Options</th>
<th>DB2 Cache Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>• DB2 caching is preferable if central and expanded storage is available</td>
<td>• Large dedicated VBP</td>
<td>• Hiperpool to back virtual buffer pool</td>
</tr>
<tr>
<td>Med. Cache Read Hit Percentage (20 to 50%)</td>
<td>• Larger disk cache</td>
<td>• More VBP/HP/GBP if high access intensity</td>
</tr>
<tr>
<td>• “MAY” storage class</td>
<td>• Record cache</td>
<td>• Isolate and study DB2 statistics</td>
</tr>
<tr>
<td>Low Cache Read Hit Percentage (0 to &lt;20%)</td>
<td>• Fastest disk drives</td>
<td>• More HP if high GP &amp; disk access intensity</td>
</tr>
<tr>
<td>• “MAY” storage class</td>
<td>• Record cache</td>
<td>• Small VBP and no hiperpool if not</td>
</tr>
</tbody>
</table>

Given the SMF 46-2 reports, look at the high access intensity page sets and analyze their cache read hit%. High cache read hit% are greater than 50%, medium, 20 to 50% and low is less than 20%. A page set with a consistently high cache read hit percentage has a high degree of re-referencing the same set of pages, and is a good candidate for improved DB2 buffering. Increase the size of the virtual buffer pool–hiperpool used with these page sets. If the I/O rate is extremely high, a dedicated buffer pool may be worthwhile.

A page set consistently in the medium hit percentage range may have varying degrees of re-reference caused by a mix of access paths used over a variety of applications. The page set may benefit from additional buffering resources either within DB2 or the cache control unit. Use DCME and/or Record Cache I with second generation controllers.

A low read percentage page set implies the lowest degree of re-referencing--more toward true random processing—or exceptional DB2 buffering. If the access intensity is high, it is not a result of good buffering. Low read hit% page sets should not share DB2 buffering space with higher percentage page sets. Put them in a separate pool, track the buffer pool internal random hit ratios versus the pool size and adjust accordingly. It is unlikely that a hiperpool will help. If the hit rate does not improve significantly by increasing the size, decrease the size and give the extra space to high hit% page sets.

To improve external access time to low read hit% page sets, place them on the “fastest” disk volumes. DCME or Record Cache in second generation controllers can be used to ensure the data pages remain cache resident longer to increase the probability of a hit during periods of higher activity.

© Compuware Corporation, 1999 Tuning External I/O in DB2 for OS/390 10
Where to Apply Additional Resource Based on Disk Cache Hit Percentage

<table>
<thead>
<tr>
<th></th>
<th>High</th>
<th>Medium</th>
<th>Low</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Buffer Pool</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hiperpool</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Group Buffer Pool</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>More Controller Cache</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Record Cache (RC)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCME (2nd Generation)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>“Fastest” Disk</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Third Generation Controllers Are The BEST Option

This chart summarizes the features that can be used with page sets based on their controller cache hit percentage to improve their access performance.

Page sets with consistently high hit percentages in the controller cache benefit by using more DB2 virtual buffer pool and hiperpool space. If sufficient central and expanded storage is available, virtual buffer pools and hiperpools may also improve the performance of page sets with medium hit percentages.

Investing more in central and expanded storage versus disk cache storage is an issue. When the cost of cache storage was much less than mainframe extended storage, the installation might favor disk cache despite the 50 times slower performance. As older bipolar storage technology is replaced by CMOS, the new costs of central, expanded and controller cache storage is roughly equal. Then it becomes a question of the limitations of the hardware and software. The latest IBM CMOS machines offer as much as 32 gigabytes of “central” storage even though the OS/390 operating system is limited to 31-bit addressing or two gigabytes. Storage beyond two gigabytes is treated logically as expanded storage and is used for hierspaces, dataspaces (with DB2 Version 6) and hiperpools. Except for massive parallel sequential processing, central storage always outperforms cache storage.

Lower hit percentage page sets are accessed more randomly. They do not gain as much from DB2 buffering or disk caching technologies except for very small page sets of just a few pages each. These can be “tucked into” a relatively small virtual buffer/hiperpool combination and incur no disk overhead once the pages are fetched into the pool. Large page sets, often the heart of a major enterprise’s database, frequently have too low a re-reference pattern to benefit much from caching. Different disk caching options may help. But often the only solution to good performance is to use the fastest random access devices available in the installation for these page sets.

Turning to examples using the analysis methodology, this is a list of just three critical DB2 index page sets on 3390 disks in an installation using second generation disk controllers.

Page sets DS01 and DS02 have greater than acceptable access intensity values. Volumes VOL001 and 002 are obviously overloaded with work. This shows up in the QUE and total RESPONSE times. DISConnect time is also high, suggesting the channel path is also overloaded and rotational position sensing interrupts are not being fielded fast enough to catch the physical block as it passes the read/write head. These volumes would be overloaded even if no other page sets shared space on them.

Both DS01 and DS02 show a great deal of update because of the low R/W ratios. Both are doing fairly well with their read hit percentage and extremely well with the DFW (deferred write) hit percentage. Controller cache is helping but cannot provide enough cache space to avoid the very poor service times. Only more DB2 virtual buffer pool and hiperpool space can help. An isolated pool for the two page sets may be useful because of their high-update activity. The DB2 buffer pool thresholds for write (VDWQT) might be increased to reduce the number of write operations decreasing the I/O load.

Page set DS03 is managed by DFSMS in the “MAY” storage class. This index has an access intensity of 0.054 so it does not appear high on the list of key page sets. However, the DBA knows that this is a critical index and should be heavily accessed. One reason the external access intensity rate is low is that the DB2 buffer manager is realizing a good retrieval hit ratio within the virtual buffer pool. Although fairly well buffered, the page set still is averaging 4.63 I/Os per second, has a very high R/W ratio and a cache hit rate indicating a high degree of page re-reference. Therefore, more DB2 buffers should further improve performance.

Upgrading to a third generation disk controller is also an attractive option. But is that tuning?
Because DS03 is so critical, its service time components and I/O rate were graphed for closer study. As often happens, an average over a long period obscures peak processing periods. This index has a higher I/O rate before noon and is very light after 2 p.m. Before noon access is purely random. The service time is better than is normally expected with a 3390 disk. The low average connect time (1.6 ms) implies most of the I/Os were synchronous for single 4K-pages. Sequential prefetch asynchronous operations have much longer connect times reading thirty-two pages in the one channel program.

After noon and into the evening, batch processing is used more heavily. Between seven and eight p.m., the access profile shows significant queue time and longer disconnect time. This indicates two changes. First, the volume is being much more heavily accessed for page sets other than this index. Second, the access to this index has a much higher contribution from sequential access. The heavy batch processing on this volume appears to be running at a higher priority than DB2 threads, as indicated by the long queue time. Another explanation is that the other work accessing the volume is a DB2 utility. When DB2 utilities read they use sequential prefetch with channel programs that access 64 pages at a time instead of the 32 for an application thread. The characteristics of these competing threads and their page sets access pattern has a major impact on the contents of cache storage in the disk controller.

This report shows the top four DB2 pagesets from a second installation. Due to good DB2 buffering, none had a high external access intensity. Two had moderate cache read hit percentages while two had very low percentages. DCME inhibits were observed for DS01 and DS03 but not for DS02 and DS04. This is because the cache controller for DS02 and DS04 did not have extended platform microcode and did not support DCME. Record cache was also not available on that controller. The page sets are predominately retrieval rather than update as shown by the high read/write ratios.

Three of these pagesets represent the most heavily accessed partitions of a very large table. DS01 is the only pageset that is processed sequentially (indicated by the average connection time of 26.2 milliseconds transferring multiple pages with sequential prefetch).

The DB2 SEQCACHE option was set to NO. This results in the SEQ % column showing that the buffer manager did not request disk controller caching for its prefetch operations to any measurable extent.

Throughout the period (6 in the evening until 7 the next morning), the partitions of the partitioned table space displayed random accessing characteristics with poor locality of reference. The highest hit rate was 38.7% and two were below 6%. During nine of the 13 hours, the hit rate was close to just one per cent. This random profile is illustrated in the chart on the next page by the short connect times indicating a predominance of single 4K-page transfers and long disconnect times during the physical access arm movement of the seeks and the rotational delay. All in all, the access patterns indicate that there is little value in using more DB2 buffering space or control unit caching.
Because the time frame is a period where batch is expected to be the predominate type of processing, these page sets are more typical of large tables with purely random processing for on-line transactions. This should be discussed with the application owners with respect to the physical database implementation. Are the right keys and indexes being used to access these tables? If the workload during this period is batch, can the input be sorted into a more optimum sequence? Is DB2 being asked to do awkward joins where these page sets contain tables that are the outer tables of the join and, although the inner table(s) is/are being accessed efficiently with prefetch I/Os, the outer tables are accessed randomly?

This DB2 subsystem had a total buffer pool size of 75 megabytes (18,300 pages). The original cache controller had two gigabytes of cache storage.

**Recommendations**: Move page sets DS01 and DS03 to a cached disk controller. Let controller microcode manage these page sets using the DFSMS “MAY” storage class. The controller logic will inhibit these page sets from using the cache, freeing the cache for other page sets with higher cache hit percentages. Consider putting these page sets on the fastest disk units in the shop to decrease access time. Assign the page sets and other page sets with low cache hit percentages to a separate, smaller DB2 virtual buffer pool. Because the page sets have such a low re-reference rate there is no reason to waste buffer pool space for pages that will be read and processed once and then not accessed (re-referenced) for long periods. A DB2 Version 6 FIFO buffer pool is an excellent option for these page sets.

This chart shows both the DB2 accounting statistics for a long running batch job step and a page set report for two DB2 index page sets identified as critical to the batch run. The batch job is run only once a month as a part of month-end processing. The asterisks in the R/W column indicate that the application is read-only.

Note that batch can drive very high I/O rates. DS01 averaged almost 85 accesses per second with DS02 adding another 47. Yet both have access intensity values below the performance threshold of 0.3. Further, both are on the same disk, VOL001! This would be an impossible bottleneck without cache storage on the disk controller.

The explanation of the high I/O rate is simple after examining the values for the read hit %. The cache controller is working well. High read cache hit rates of almost 100% reduce the average response time to 2.8 milliseconds with connect time of only one millisecond. The disk cache is earning its keep by holding a large amount of prestaged data that is being heavily re-referenced.

But wait! The accounting statistics show that the process is heavily I/O bound (only 2 hours of DB2 processing out of 14 hours of wall clock time). Further DB2 is spending most of the 14 hours waiting for synchronous I/O. The controller statistics verify this by the average connect time of only one millisecond. Sequential prefetch is not being used--these are all one 4K-page synchronous reads to pages that are almost always read cache hits.
Further analysis provided a solution to this puzzling profile. The two indexes totaled around 4,900 pages. They are accessed through a virtual buffer pool of 1,500 pages (6.1 megabytes). The 3,400 pages that could not fit into the virtual buffer pool gradually became resident in the disk controller’s cache and were continually retrieved from there using external I/O requests. With the average response time of 2.8 milliseconds and an average I/O rate of 132 reads per second for the two page sets, almost 370 milliseconds of external I/O was being used per second. Had these pages been in a hiperpool, the access time would have been closer to 5.3 milliseconds (132 accesses per second at 40 microseconds). That is both a 70 to 1 improvement in access speed and a good reduction in the CPU workload needed to drive the 132 external I/Os. A larger virtual pool would provide even faster throughput.

**Recommendation:** Because the installation wanted to improve the throughput of this batch job as much as possible and central storage was no problem, a virtual buffer pool of 4,900 pages was dedicated to the two indexes. The elapsed time for the batch work was reduced from 14 to 1.5 hours moving from I/O-bound to CPU-bound. This is another example of the fact that just about any program can be made CPU-bound given enough high speed buffering space.

It is also interesting to speculate on how this program would have behaved without the plain luck of having the cache controller in the first place. Random I/Os at 12 to 20 milliseconds versus the 2.8 service time from cache suggests that the 14 hours elapsed time could have been on the order of 60. Sixty versus fourteen versus one and a half dramatically shows the importance of understanding the DB2 access hierarchy options.

**Acknowledgments:** Special thanks are extended to Mr. John Berg and Ms. Nghi Eakin of IBM, Gaithersburg, Maryland for providing the foundation for this paper. They, in turn, were supported by many IBM associates including Mr. Dave Betten, Dr. Frank Chu, Mr. Roger Miller, Mr. Martin Moroch, Mr. Jack Stanton and Dr. James Teng. Another colleague was Ms. Cindy Rollins of Lee Resources.
References:
IBM Performance Management Service Offerings Internet Home Page (URL: http://www.as.ibm.com/asus/pmos.html)
“IBM RAMAC Virtual Array,” IBM Corporation, SG24-4951, July, 1997
“Introduction to IBM Direct-Access Storage Devices and Organization Methods,” IBM Corporation, GC20-1649-10, about 1969
“Performance Management in a DFSMS World,” IBM Corporation, GG66-3252
“Understanding the DB2 Version 3 Buffer Manager,” Compuware Corporation, 1995

Trademarks:
The following trademarks of the International Business Machines Corporation are used throughout this presentation: DB2, DFSMS, DFSMS/MVS, Hiperspace, IBM, MVS/ESA, OS/390, RAMAC.