DB2 Information Management Software

IBM

Redbooks Paper

Paolo Bruni
John Iczkovits

# Disk storage access with DB2 for z/OS

Disk storage has changed rapidly over the past few years with the delivery of new functions and improved performance. DB2® has made many changes to keep pace and make use of the disk improvements:

- DB2 integrates with the storage management software and continues to deliver synergy with FICON® (fiber connector) channels and disk storage features.
- DB2 uses Parallel Access Volume and Multiple Allegiance features of the IBM® TotalStorage® Enterprise Storage Server® (ESS) and DS8000™.
- FlashCopy® V2 helps increase the availability of your data while running the CHECK INDEX SHRLEVEL(CHANGE) utility.
- DB2 integrates with z/OS® 1.5 and later to deliver the system point-in-time recovery capability, a solution applicable to recovery, disaster recovery, or environment cloning needs.
- ESS and DS8000 FlashCopy are used for DB2 BACKUP and RESTORE SYSTEM utilities, which implement the solution.
- Larger control interval sizes help performance with table space scans, and resolve some data integrity issues.
- Striping is an effective way of increasing sequential throughput by spreading control intervals across multiple devices.
- The MIDAW function, new with the z9™-109 server, improves FICON performance by reducing channel utilization and increasing throughput for parallel access streams.

In this IBM Redpaper, we explore several of these features and the improved synergy in recent versions up to DB2 for z/OS Version 8. We cover the following topics:

- Disk hardware architecture
- Disk functions
- SMS
- DSNZPARMs relating to disk storage management
- Utilities and FlashCopy
- The MIDAW facility: improved channel architecture

# Disk hardware architecture

Disk hardware has evolved significantly since IBM introduced its first direct access storage device (DASD) back in 1956, the IBM 350. Over the years, newer disk hardware resulted in the advantages of more space per device, a smaller footprint, faster throughput of data, and improved functionality. DB2 performance increases as newer disk technologies emerge. In this chapter we discuss the nuances of disk hardware architecture.

## Real, virtual, and virtualized disk

Many people still see disks from a 3380 or 3390 perspective. This architecture was considered *real* because data was accessed directly from a location where the logical and physical locations were the same.

The RAMAC® Virtual Array (RVA) was based on *virtual* disks that emulated 3380s and 3390s. With the RVA, we saw our data from a logical disk point of view of an emulated device. This means that an emulated 3380 would look like a real 3380 and an emulated 3390 would look like a real 3390. The RVA mapped our view of a logical disk to a physical location in the box. Although the RVA wrote to a physical disk, it did not require the capacity of a fully emulated device; only the actual space was required.

The ESS and the latest DS8000 use *virtualized disk*. They still emulate devices and the boxes map logical disks to physical locations, but full physical capacity is required for data.

With the RVA, ESS, and DS8000, data is dispersed among several DDMs (Disk Drive Modules). DDMs are the physical location where data resides. With virtual and virtualized disks, we need to see a disk volume from a logical perspective instead of a physical one.

Although the RVA, ESS, and DS8000 all work with logical disks that map to physical locations, they are three different architectures and cannot be viewed as the same type of hardware.

## RAID technology

RAID is the acronym for *Redundant Array of Independent Disks*. RAID is a category of disk drives combined together for fault tolerance and performance. It is an important technology to understand, even from a DB2 perspective.

The RAID technology is usually associated with eight different RAID levels, and is disk type dependent:

- RAID 0 - data striping without parity
- RAID 1 - dual copy
- RAID 2 - synchronized access with separate error correction disks
- RAID 3 - synchronized access with fixed parity disk
- RAID 4 - independent access with fixed parity disk
- RAID 5 - independent access with floating parity
- RAID 6 - dual redundancy with floating parity
- RAID 10 (DS8000 and some ESS) - RAID 0 + RAID 1, no parity

As this paper is being written, most customers use RAID 5 or 10 technology.

Parity is additional data, “internal” to the RAID subsystem, which enables a RAID device to regenerate complete data when a portion of the data is missing. Parity works on the principle that you can sum the individual bits that make up a data block or byte across separate disk drives to arrive at an odd or even sum.

## Staging data into disk cache

Cache is a very important part of DB2 performance. When we retrieve data from disk, first it is staged into cache and then transferred from cache to a buffer pool. Data access is determined through the controller's adaptive cache in the following manner:

- Random read
  - Record or block staging (page)
  - Partial track staging (from the page requested until the end of the track)
  - Full track staging

  Adaptive cache determines which of the three is used based on previous use of data whereby anything from as small as one page, or as large as from the page until the end of the cylinder plus the next cylinder, will be read in.

- Sequential prefetch
  - Full track staging

  Prefetch from the page up to the end of the cylinder plus the next cylinder (extent boundary). Actual operation is done on extent boundaries or stage group.

When we write to disk, we actually write to cache, then the disk box is eventually destaged from cache down to disk. DB2 sees the write to cache as a disk write. Using this mechanism provides great performance benefits because we do not lose time for the traditional disk latency, spin, seek, and so forth to write out data—the disk controller does it for us.

From a conceptual point of view, disk cache and buffer pools have the same function, keeping commonly used data around in an area that can be retrieved quickly.

# Disk functions

Disk architecture changes with the advent of newer disk hardware. New functionality is integrated into newer disk boxes as well as into versions of DFSMSdfp™. We need to understand not only DB2 functions, but hardware and MVS™ characteristics as well.

## Parallel access volume

Parallel access volume (PAV) enables a single zSeries® server to simultaneously process multiple I/O operations to the same logical volume, which can help to significantly reduce device queue delays (IOSQ time). This is achieved by defining multiple addresses per volume. With dynamic PAV, the assignment of addresses to volumes can be automatically managed to help the workload meet its performance objectives and reduce overall queuing.

With PAV, reads are simultaneous. Writes to different domains (a set of tracks the disk controller is working on) are simultaneous as well. However, writes to the same domain are serialized. No double updates are possible to preserve integrity.

Large volumes such as 3390 mod 9, 27, and 54 greatly benefit by using PAV.

Multiple paths or channels for a volume have been around for many years, but multiple UCBs (MVS addresses) were only introduced with PAVs. See Figure 1.

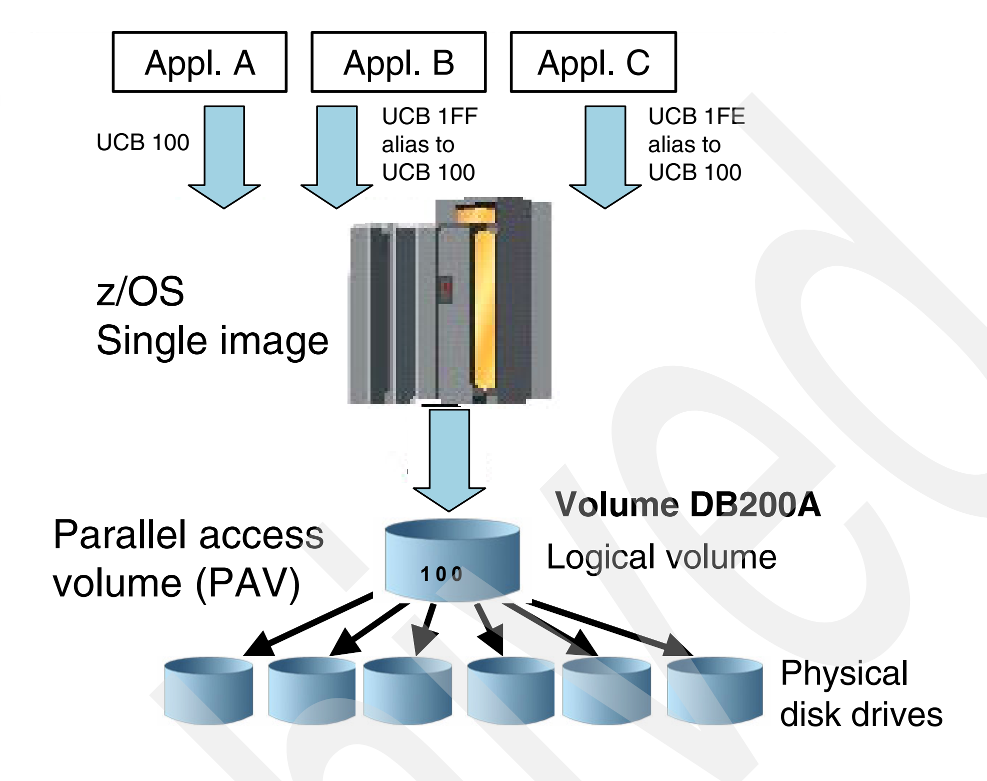


*Figure 1 Parallel access volume (PAV)*

Accessing data from one logical volume in today's world requires accessing several DDMs. DDMs are the physical location where data resides. For example, although my table space called SUEICZ resides on logical volume DB200A, the data in modern disk is striped across several DMMs or physical drives.

- In this example, from a DB2 perspective, applications A, B, and C can be replaced with threads 1, 2, and 3, or even a mix such as thread 1, thread 2, and Appl. 1. The point is that there are multiple requests for access for this volume.
- The logical device 100 is the UCB (Unit Control Block), the MVS address for our volume DB200A. There are three requests for access to DB200A, the base address is 100, and the aliases to this volume are 1FE and 1FF. The number of aliases depends on how your MVS Systems Programmer assigned the device. When using dynamic PAV, which is the recommendation, a heavily accessed DB2 volume with many data sets may have several PAVs assigned.
- With volumes attached to multiple LPARs, every LPAR is required to use the same number of PAVs for each volume. Some system overhead is present when dynamic PAV is moving the aliases for handling dramatically different workloads across the multiple APARs.

## Multiple allegiance

Multiple allegiance (MA) expands the simultaneous logical volume access capability across multiple zSeries servers. This function, along with PAV, enables the ESS and DS8000 series to process more I/Os in parallel, helping to improve performance and facilitating use of large volumes. See Figure 2.

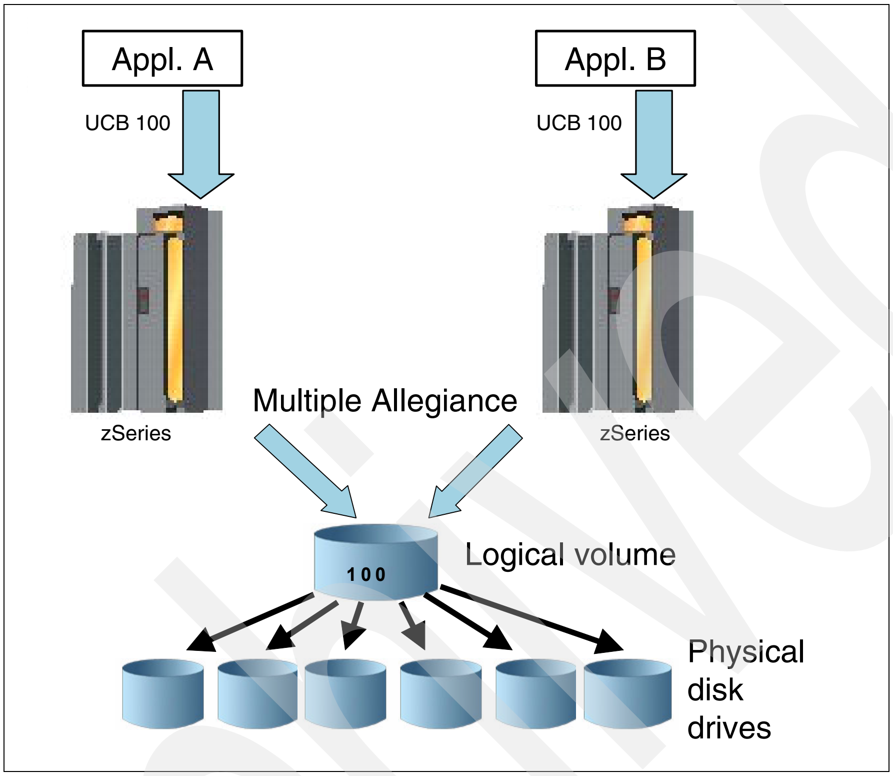


*Figure 2 MA (multiple allegiance)*

MA reduces PEND time (see Figure 4 on page 9) and is especially useful in data-sharing environments.

MA allows for access of a logical volume by more than one LPAR where PAVs only allow access to a logical volume from one LPAR.

## VSAM data striping

VSAM data striping allows sequential I/O to be performed for a data set at a rate greater than that allowed by the physical path between the disk and the processor. The physical characteristics of channels, control units, and disk limit the data transfer rate. VSAM data striping avoids such limitations by spreading the data set among multiple stripes on multiple control units. This is different from most of the RAID technologies that stripe data in the same disk array. The data striping function is designed to improve the performance of applications requiring sequential access to data records. Data striping does not affect direct access to data.

An equal amount of space is allocated for each stripe. If the guaranteed space attribute is used in the storage class, the specified quantity is allocated to each volume in the stripe count. For a data set with the non-guaranteed space attribute in the storage class, the initial allocation quantity is divided across all volumes in the stripe count.

Striped VSAM data sets are in extended format (EF) and are internally organized so that control intervals (CIs) are distributed across a group of disk volumes or stripes. A CI is contained within a stripe.

DB2 V8 enables all pages sizes to be striped when DSNZPARM value for DSVCI is set to YES. Refer to APAR PQ53571 for issues with pages greater than 4 KB prior to DB2 V8.

VSAM data striping should be used only for those non-partitioned objects where the majority of activity is heavy sequential processing. Recommendations about where to use VSAM data striping because of their highly sequential nature include:

- DB2 active log data sets
- DB2 work files
- Segmented table spaces and NPIs that are mostly and heavily sequential
- LOBs

## Sequential data striping

As well as being beneficial to stripe a linear data set (LDS) that is heavily sequential, it is also beneficial to stripe, whenever possible, the input and output files for utilities that are associated with a striped table space. Examples of what would be beneficial to stripe when the associated table space is striped include:

- Image copy data sets
- Input to LOADs
- Utility work files

Only disk data sets can be striped.

## Cylinder and head information

The physical location of your data no longer resides at the cylinder and head (CCHH). From our view we see data for a logical volume from the VTOC perspective only. The disk box itself maps data between the VTOC and physical locations. Be aware that because we see data from a VTOC perspective, messages and information are from a CCHH perspective.

Messages still show CCHH information for virtual and virtualized disk (RVA, ESS, and DS8000). See Example 1.

*Example 1 CCHH output*

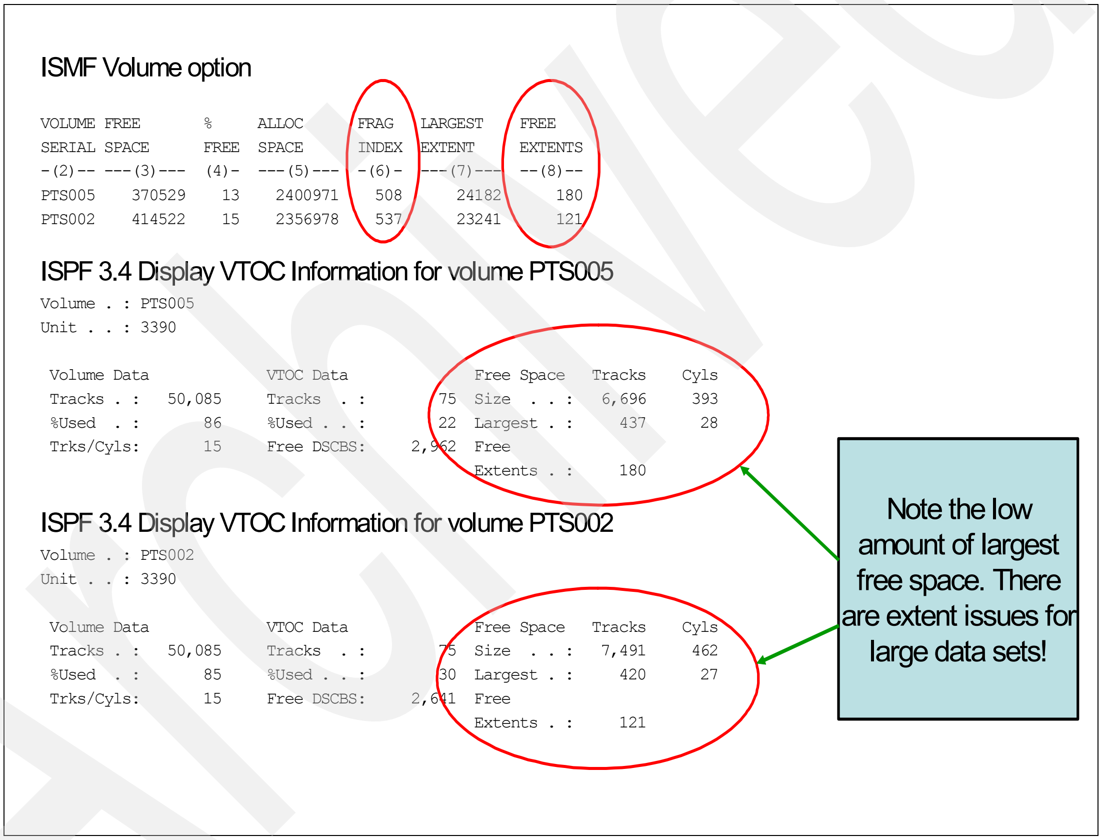
```
LISTCAT output
EXTENTS:
LOW-CCHH-----X'079D0000'
HIGH-CCHH----X'07AC000E'

IEHLIST utility with FORMAT option
EXTENTS  NO  LOW(C-H)   HIGH(C-H)    NO  LOW(C-H)   HIGH(C-H)
        0      0  1        0  1      1     0 12        0 14

DSNU538I RECOVER ERROR RANGE OF DSN=dataset name ON VOLUME=volser FROM
CCHH=X’cccchhhh’ TO CCHH=X’cccchhhh’ CONTAINS PHYSICAL ERROR
```

## Volume fragmentation

In the DB2 environment, frequent CREATEs and DROPs of table spaces and indexes cause volume fragmentation. This is typically even more evident in test and development environments. High fragmentation results in:

- Inefficient use of disk space
- An increase in space-related abends
- Performance degradation caused by excessive disk arm movement
- An increase in the time required for functions that are related to direct access device space management (DADSM)

The Interactive Storage Management Facility (ISMF) is one of the tools a Storage Administrator uses to manage the storage environment. Figure 3 shows the ISMF output of the Volume option.



ISMF Volume option

```
VOLUME FREE       %    ALLOC     FRAG  LARGEST   FREE
SERIAL SPACE      FREE SPACE     INDEX EXTENT    EXTENTS
-(2)-- ---(3)---  (4)- ---(5)--- -(6)- ---(7)--- --(8)--
PTS005    370529   13   2400971   508    24182      180
PTS002    414522   15   2356978   537    23241      121
```

ISPF 3.4 Display VTOC Information for volume PTS005

```
Volume . : PTS005
Unit . . : 3390

 Volume Data           VTOC Data               Free Space   Tracks    Cyls
 Tracks . :   50,085   Tracks   . :         75 Size   . . :  6,696     393
 %Used  . :       86   %Used . . :          22 Largest . :     437      28
 Trks/Cyls:       15   Free DSCBS:       2,962 Free
                                               Extents . :     180
```

ISPF 3.4 Display VTOC Information for volume PTS002

```
Volume . : PTS002
Unit . . : 3390

 Volume Data           VTOC Data               Free Space   Tracks    Cyls
 Tracks . :   50,085   Tracks   . :         75 Size   . . :  7,491     462
 %Used  . :       85   %Used . . :          30 Largest . :     420      27
 Trks/Cyls:       15   Free DSCBS:       2,641 Free
                                               Extents . :     121
```

Note the low amount of largest free space. There are extent issues for large data sets!

*Figure 3 Fragmentation page*

We need to pay close attention to columns 6 (fragmentation index) and 8 (free extents) of the ISMF Volume option.

Fragmentation index can go as high as 999. As the fragmentation index increases, you see more space-related problems. What the actual value for fragmentation index should be is arguable, but a good strategy is to start DEFRAG(ment)ing volumes above a fragmentation index of 250.

Volume PTS002 has 462 cylinders available, but the largest available free extent is only 27 cylinders. If you allocate a 250-cylinder data set on PTS002, which is just a little more than half of the space available on the volume, you will see the following results:

- SMS allocation: It depends (space relief is not valid for multiple striped data sets):
  - Failure again if the guaranteed space attribute is set to YES with no space constraint relief. Same scenario as above.
  - Success when the guaranteed space attribute is set to NO and there are other volumes in the storage group that can accommodate the allocation; otherwise, failure.
  - Success when the guaranteed space attribute is set to YES with space constraint relief on. However, you might run out of extents trying to get the 250 cylinders if the allocation is for a non-EF data set.
- Non-SMS allocation: Failure dealing with the five extents for primary allocation rule, even if there are five extents of 27 cylinders; the result would be:
  - 5 extents * 27 cylinders = 135 cylinders. For the primary allocation, just a little more than half the space required. Running DEFRAG on this volume would probably combine enough free extents to accommodate the space request.

The solution to the fragmentation problem is to run the DFSMSdss™ utility DEFRAG. DEFRAG consolidates free space on volumes by relocating data set extents on a volume.

Storage Administrators can use data set FlashCopy V2 for ESS and DS8000 devices or SnapShot for RVAs to provide much faster DEFRAG operations. Storage Administrators can also use the FRAGI keyword, which only DEFRAGs volumes above a specific fragmentation index. FRAGI(250) would be a good place to start. After some time you might want to lower the value for FRAGI or totally eliminate the keyword altogether.

For some customers, the time to run DEFRAGs might be an issue because it is disruptive to all other work on the logical volume. In this case consider running DEFRAG while doing maintenance when DB2 is down. Keep these requirements in mind when running DEFRAG:

- DEFRAG processing locks the VTOC (through the RESERVE macro) and the VVDS. The DEFRAG function also serializes on data sets through ENQ or dynamic allocation. Data set access is severely restricted.
- In order to run DEFRAG, the volume must be offline to all LPARs except for the one running the DEFRAG.

## RMF volume reports

RMF™ is the z/OS Resource Measurement Facility. It generates a number of standard reports including CPU and direct access device activity reports. Storage Administrators review RMF reports on a periodic basis in order to determine how well the disk devices your DB2 data resides on are performing. From an RMF perspective, there are four stages of I/O operation (shown in Figure 4 on page 9) that a Storage Administrator compares to the overall time.

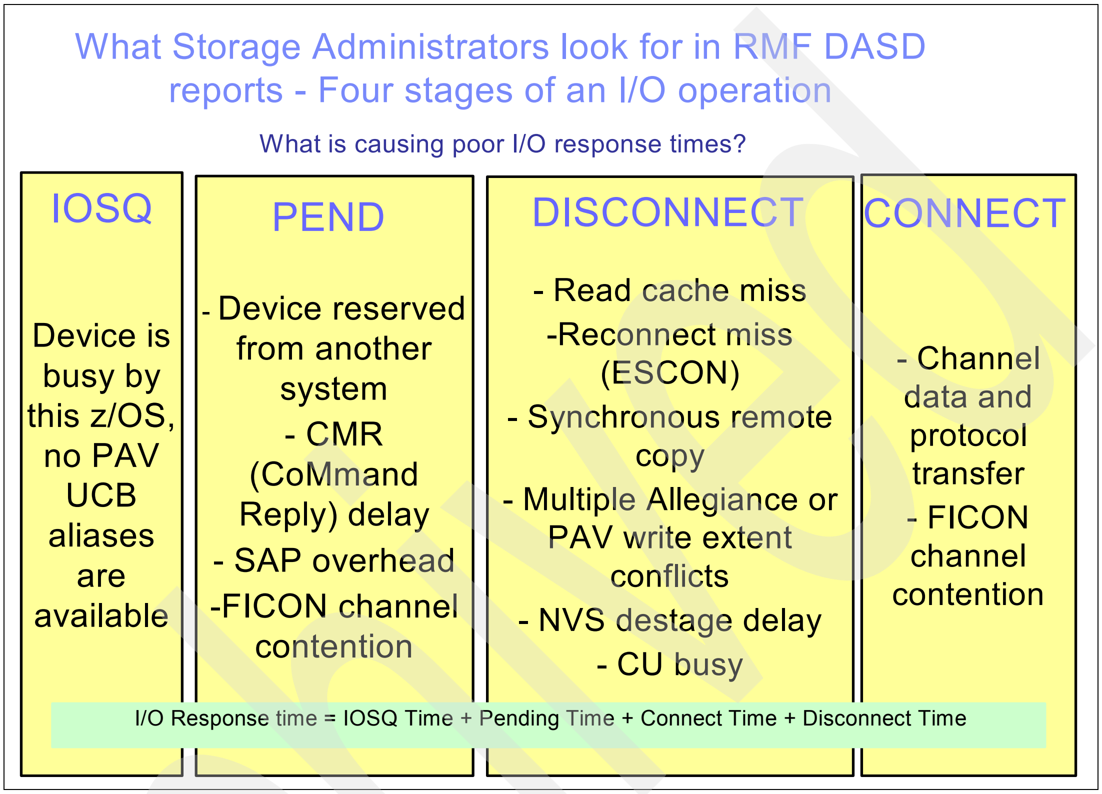
**IOSQ** Time waiting for the device availability in the z/OS operating system (UCB time). PAV can reduce this time. High IOSQ time can generally be resolved if there are enough PAV UCBs defined.

**Pending** Time from the SSCH instruction (issued by z/OS) till starting the dialog between the channel and the I/O controller. It is related to channel, part, control unit, and device busy. Multiple allegiance (MA) can reduce this time.

**Disconnect** Time that the I/O operation already started but the channel and I/O controller are not in a dialog.

**Connect** Time when the channel is transferring data from or to the controller cache or exchanging control information with the controller about one accepted I/O operation. Using FICON, this is a measure of physical disk access.

Total response time is the sum of these four fields.



What Storage Administrators look for in RMF DASD reports - Four stages of an I/O operation

What is causing poor I/O response times?

IOSQ

Device is busy by this z/OS, no PAV UCB aliases are available

PEND

- Device reserved from another system
- CMR (CoMmand Reply) delay
- SAP overhead
-FICON channel contention

DISCONNECT

- Read cache miss
-Reconnect miss (ESCON)
- Synchronous remote copy
- Multiple Allegiance or PAV write extent conflicts
- NVS destage delay
- CU busy

CONNECT

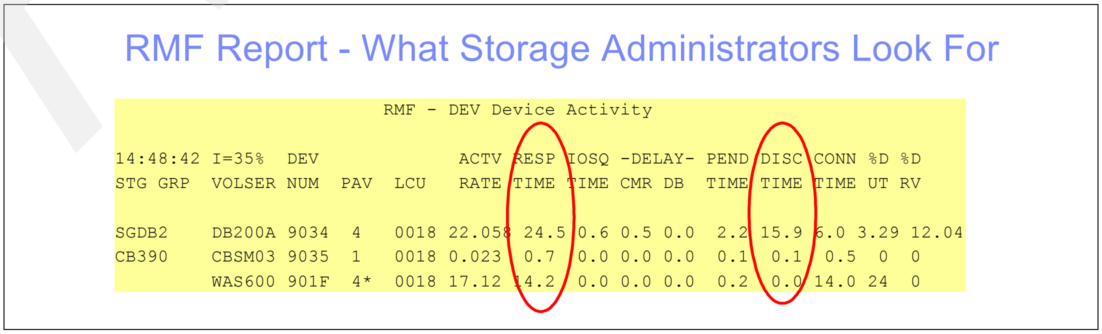
- Channel data and protocol transfer
- FICON channel contention

I/O Response time = IOSQ Time + Pending Time + Connect Time + Disconnect Time

*Figure 4 Stages of an I/O operation*

Hot spots still occur, even in modern disks. In the report shown in Figure 5, disconnect time is the culprit. Several problems could cause high disconnect time relative to response time.

One of the more common causes, as in our example, is where a logical subsystem (LSS) was front-loaded and is overcommitted. This happens when a Storage Administrator assigns volumes in UCB (MVS address) order without spreading volumes across control units, thereby causing too much work for a disk controller to handle. One solution is to move the volume to a less heavily used logical control unit (LSS or LCU); another is to move heavily used data sets to a less heavily used LSS/LCU. This scenario can have a negative impact on DB2 performance. Resolving this problem can enable DB2 to perform within proper limits.



RMF Report - What Storage Administrators Look For

```
                         RMF - DEV Device Activity

14:48:42 I=35%  DEV              ACTV RESP IOSQ -DELAY- PEND DISC CONN %D %D
STG GRP  VOLSER NUM  PAV  LCU    RATE TIME TIME CMR DB  TIME TIME TIME UT RV

SGDB2    DB200A 9034  4   0018 22.058 24.5 0.6 0.5 0.0  2.2 15.9 6.0 3.29 12.04
CB390    CBSM03 9035  1   0018 0.023   0.7 0.0 0.0 0.0  0.1  0.1  0.5  0   0
         WAS600 901F  4*  0018 17.12  14.2 0.0 0.0 0.0  0.2  0.0 14.0 24   0
```

*Figure 5 RMF report - High disconnect time*

## Disk response time ballpark numbers

DB2 response times when dealing with disks are faster with each generation of new disks. Table 1 lists an example of ballpark disk response times. The table shows sequential read or write with cache hits versus random reads without cache calculated for a single page (not sequential disk response times, but disk response times divided by the number of pages for the common sequential number of pages per IO, 32 or 4).

Today's large caches often get 90% to 95% cache hits.

*Table 1 Ballpark I/O times*

| | **Sequential read or write** | | **Random read** | |
|---|---|---|---|---|
| | **4 KB page** | **32 KB page** | **4 KB page** | **32 KB page** |
| 3390, RAMAC1, RAMAC2 | 1.6 to 2 ms | 14 ms | 20 ms | 30 ms |
| RAMAC3, RVA2 | 0.6 to 0.9 ms | 6 ms | 20 ms | 30 ms |
| ESS E20 ESCON® | 0.3 to 0.4 ms | 3 ms | 10 ms | 15 ms |
| ESS F20 ESCON | 0.25 to 0.35 ms | 2 ms | 10 ms | 15 ms |
| ESS F20 FICON | 0.13 to 0.2 ms | 1.5 ms | 10 ms | 15 ms |
| ESS 800 FICON | 0.09 to 0.12 ms | 0.5 to 1 ms | 5 to 10 ms | 10 to 15 ms |
| DS8000 | 0.035 to 0.06 ms | 0.3 to 0.5 ms | 5 to 10 ms | 5.5 to 11 ms |

Performance notes:

- *Sequential* is based on prefetch whereby one I/O can read in multiple pages. Pages are contiguous with no skip sequential. Sequential numbers are also based on pages being in cache, but *random* reads from disk.
- For 8 KB and 16 KB pages, interpolate from 4 KB and 32 KB page numbers.
- For skip sequential read or write (reading or writing of pages that are not contiguous), in the case of infrequent cache hits, the time would be somewhere between sequential and random and depends on the distance between pages read or written.
- I/O time for sequential read or write is prorated on a per-page basis, because multiple pages can be read or written by one start I/O.
- Read I/O time tends to be faster than write I/O; that is, use 1.6 to 1.8 ms sequential read and 1.8 to 2 ms sequential write for 3390.
- Random read I/O time would go down with cache hit. From 5 to 10 ms to 0.2 ms on the DS8000, that is 25 to 50 times faster.

Formula:

- Time per page is obtained by dividing an average wait time for one prefetch or deferred write I/O from accounting class 3 by the number of pages read or written. The “Sequential read or write” column represents the case in which all pages read or written are contiguous, not skip sequential.

Newer disk technology provides better performance. FICON (fiber channels) outperforms ESCON because of their faster data rates (2 Gb and 4 Gb).

The numbers that are provided are not exact. If your performance numbers are nowhere near the ballpark numbers, it is time to speak to your Storage Administrator or Performance team.

Review the RMF data with them to learn why your numbers are so skewed. This may be a lack of, or not enough, PAVs or perhaps not enough paths to devices. Your Storage Administrator or Performance team should be able to provide some detail about how the performance can be resolved in order to have DB2 perform as designed.

## Disk emulation

Because modern physical disks (DDMs) come in very large sizes, you can emulate a logical 3390 to fit VTOCs of varying sizes. There are advantages and disadvantages to choosing a particular size disk. Table 2 summarizes the emulations option.

*Table 2 Disk emulation*

| Model | Cylinders | Tracks | Bytes/volume | Bytes/track |
|---|---|---|---|---|
| 3390-1 | 1113 | 16695 | 946 MB | 56664 |
| 3390-2 | 2226 | 33390 | 1.89 GB | 56664 |
| 3390-3 | 3339 | 50085 | 2.83 GB | 56664 |
| 3390-9 | 10017 | 150255 | 8.51 GB | 56664 |
| 3390-27 | 32760 | 491400 | 27.84 GB | 56664 |
| 3390-54 | 65520 | 982800 | 55.68 GB | 56664 |

The 3390 model 1 starts with 1,113 cylinders and all models beyond it are multipliers of a mod 1. For example, a mod 2 is twice the size of a mod 1, a mod 3 three times the size of a mod 1, and so forth. The only anomaly is the mod 27, which is not exactly 27 times a mod 1. The mod 27 is 27 times a mod 1 plus 2,709 cylinders. The mod 54 is twice the size of a mod 27, therefore it too is not an exact multiplier of a mod 1.

Storage Administrators refer to 3390 mod 9, 27, and 54 as *large-volume* because they support 10017 cylinders, 32760 cylinders, and 65520 cylinders respectively. Storage manuals and articles commonly call mod 27 and 54 devices *large* or extra large *devices*.

Bytes per track refers to the device capacity only. The actual allocation for a DB2 LDS is 48 KB, not the total of 56 KB. This allows for twelve 4 KB DB2 pages to be allocated per track.

## Comparing response times for combined emulated devices

In order to determine how well larger-sized volumes would compare in performance to smaller volumes, a test was performed whereby the data from sixty 3390 mod 3s were combined down to six 3390 mod 27s. The results are evident in Figure 6 on page 12.

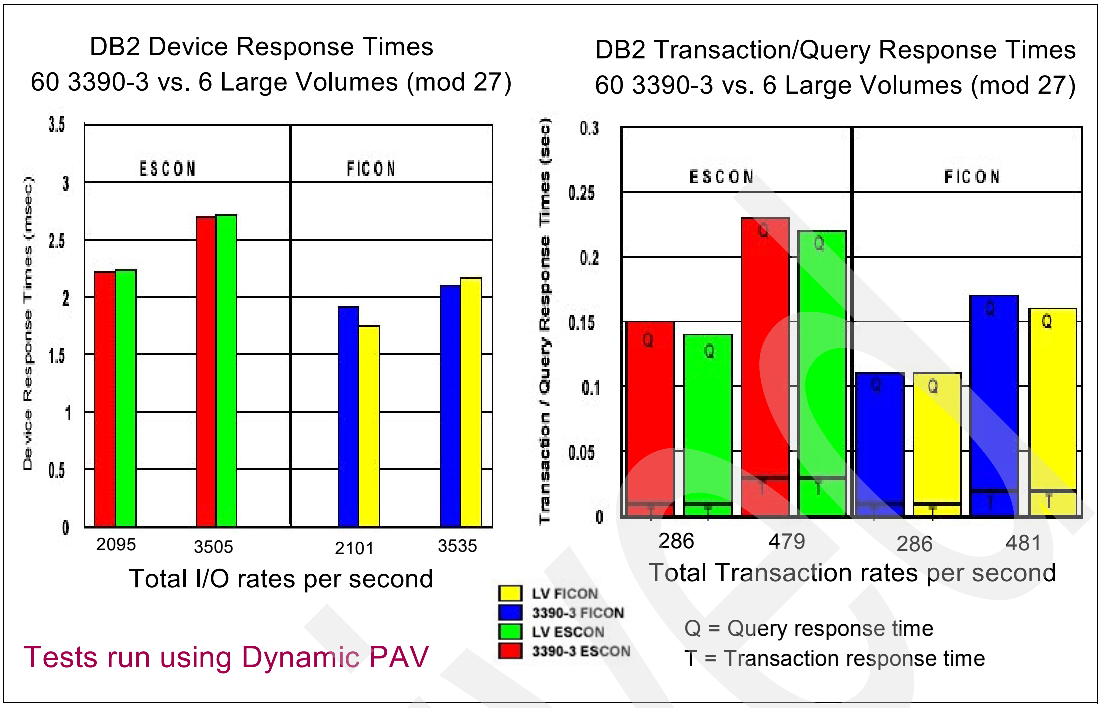


*Figure 6 Comparing response times for combining emulated devices*

Two of the tests performed included comparing:

- DB2 device response times
- DB2 transaction/query response times

Both volumes were tested with ESCON and FICON connections. There was no surprise that FICON performed better than ESCON.

What the test did show is that using 60 mod 3 devices versus six mod 27 devices with the same data did not cause a significant change in performance. Dynamic PAVs are the reason why performance was not affected when combining all of the data.

Dynamic PAVs enable you to combine table spaces, indexes, and partitions onto the same volume without major performance impacts. Although hot spots are still possible, they are less likely when using dynamic PAVs.You may choose to continue separating your table spaces and indexes into separate storage groups, but combining table spaces and indexes into one storage group on the same set of volumes is much less of an issue when using dynamic PAVs.

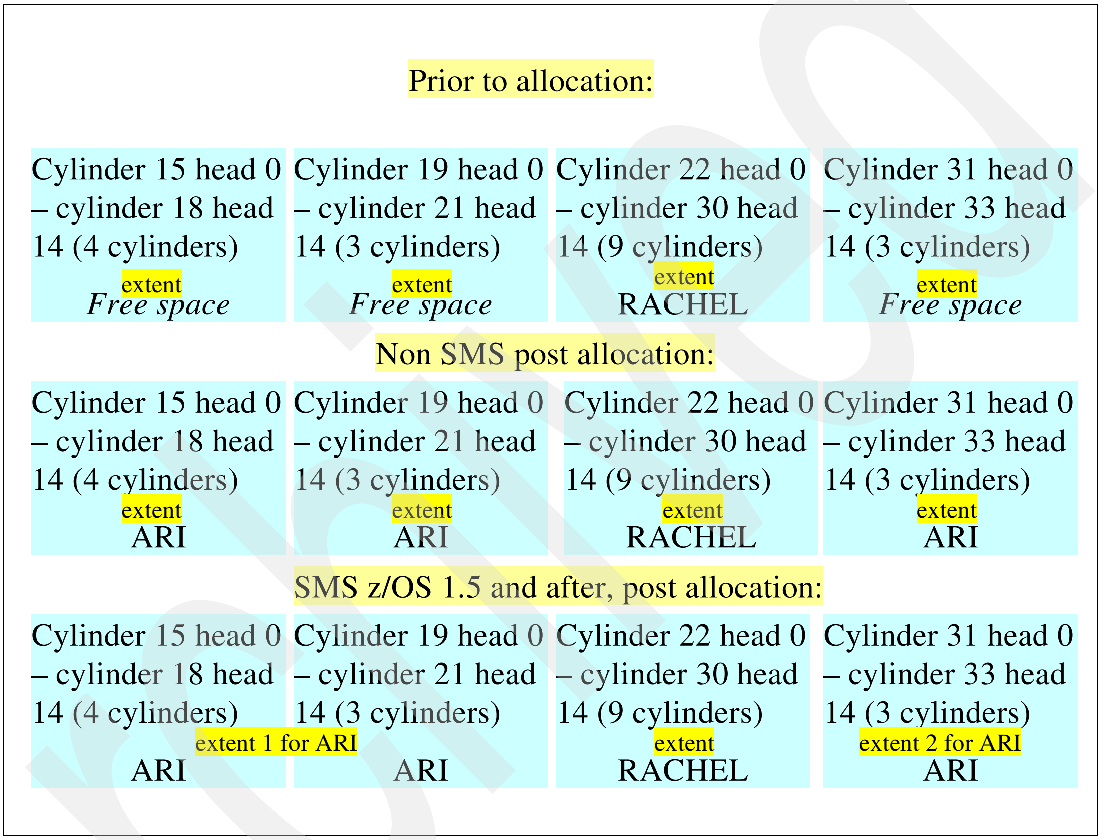
## Extent consolidation

z/OS consolidates adjacent extents for VSAM SMS managed data sets when extending on the same volume. VSAM extent consolidation is automatic and requires no action on your part. If the extents are adjacent, the new extent is incorporated into the previous extent. This means that first extent tracks as seen on ISPF 3.4 Data set level listing shows allocations larger than the primary when extents are adjacent to the primary (Figure 7 on page 13 and Figure 8 on page 14).

There is a DFSMSdfp (Data Facility Program) limit of 123 extents per volume. Extent consolidation continues even when an LDS exceeds what would have been 123 extents. For example, the allocation is PRIQTY 720 SECQTY 720, which is 1 cylinder primary and secondary and *no other data sets reside on that volume*. If you insert 190 cylinders worth of data, you will not stop at 123 cylinders for the volume, but allocate the 190 cylinders and have one extent.

VSAM LDSs that are used for DB2 must be SMS managed for extent consolidation to take effect. Extent consolidation is not used for non-SMS managed data sets, in which case the extent allocation works as before.

When allocating a 10-cylinder primary, if fragmentation is such that the data set needs to use three extents for a new VSAM object (ARI) broken down into four cylinders, three cylinders, and three cylinders, the result in the VTOC is as shown in Figure 7.



Prior to allocation:

| | | | |
|---|---|---|---|
| Cylinder 15 head 0 – cylinder 18 head 14 (4 cylinders)<br>extent<br>*Free space* | Cylinder 19 head 0 – cylinder 21 head 14 (3 cylinders)<br>extent<br>*Free space* | Cylinder 22 head 0 – cylinder 30 head 14 (9 cylinders)<br>extent<br>RACHEL | Cylinder 31 head 0 – cylinder 33 head 14 (3 cylinders)<br>extent<br>*Free space* |

Non SMS post allocation:

| | | | |
|---|---|---|---|
| Cylinder 15 head 0 – cylinder 18 head 14 (4 cylinders)<br>extent<br>ARI | Cylinder 19 head 0 – cylinder 21 head 14 (3 cylinders)<br>extent<br>ARI | Cylinder 22 head 0 – cylinder 30 head 14 (9 cylinders)<br>extent<br>RACHEL | Cylinder 31 head 0 – cylinder 33 head 14 (3 cylinders)<br>extent<br>ARI |

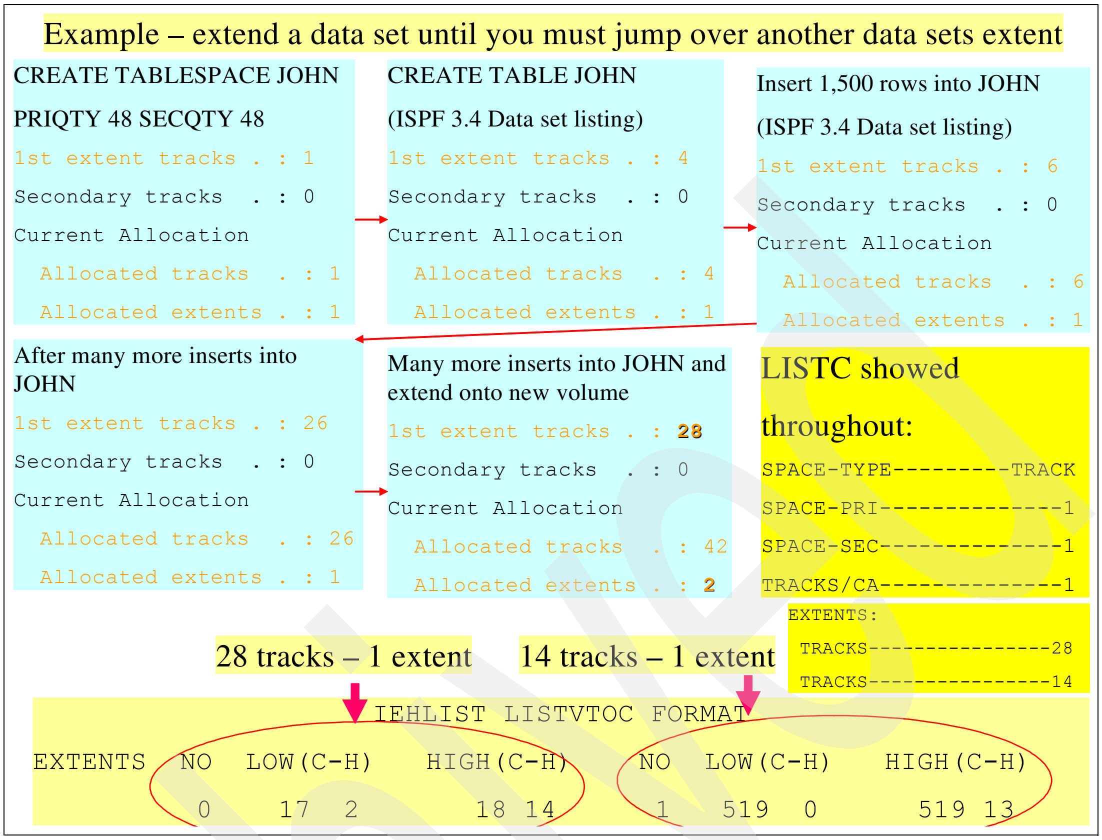
SMS z/OS 1.5 and after, post allocation:

| | | | |
|---|---|---|---|
| Cylinder 15 head 0 – cylinder 18 head 14 (4 cylinders)<br>ARI | extent 1 for ARI<br>Cylinder 19 head 0 – cylinder 21 head 14 (3 cylinders)<br>ARI | Cylinder 22 head 0 – cylinder 30 head 14 (9 cylinders)<br>extent<br>RACHEL | Cylinder 31 head 0 – cylinder 33 head 14 (3 cylinders)<br>extent 2 for ARI<br>ARI |

*Figure 7 Extent consolidation: VTOC view*

Starting with z/OS 1.5, LDSs began to use extent consolidations. As long as the free space on the VTOC is adjacent, the extents are combined into one extent. In this case new data set ARI was able to combine the first two extents, making the first extent seven cylinders and one extent. Data set RACHEL occupied the next extent, which was not free, so the second extent for ARI was the next available free space after RACHEL.

In Figure 8 on page 14 we create a table space with the space specifications of one track primary and one track secondary.



Example – extend a data set until you must jump over another data sets extent

CREATE TABLESPACE JOHN

PRIQTY 48 SECQTY 48

```
1st extent tracks . : 1
Secondary tracks  . : 0
Current Allocation
  Allocated tracks  . : 1
  Allocated extents . : 1
```

CREATE TABLE JOHN

(ISPF 3.4 Data set listing)

```
1st extent tracks . : 4
Secondary tracks  . : 0
Current Allocation
  Allocated tracks  . : 4
  Allocated extents . : 1
```

Insert 1,500 rows into JOHN

(ISPF 3.4 Data set listing)

```
1st extent tracks . : 6
Secondary tracks  . : 0
Current Allocation
  Allocated tracks  . : 6
  Allocated extents . : 1
```

After many more inserts into JOHN

```
1st extent tracks . : 26
Secondary tracks  . : 0
Current Allocation
  Allocated tracks  . : 26
  Allocated extents . : 1
```

Many more inserts into JOHN and extend onto new volume

```
1st extent tracks . : 28
Secondary tracks  . : 0
Current Allocation
  Allocated tracks  . : 42
  Allocated extents . : 2
```

LISTC showed throughout:

```
SPACE-TYPE---------TRACK
SPACE-PRI--------------1
SPACE-SEC--------------1
TRACKS/CA--------------1
  EXTENTS:
   TRACKS----------------28
   TRACKS----------------14
```

28 tracks – 1 extent

14 tracks – 1 extent

```
            IEHLIST LISTVTOC FORMAT
EXTENTS  NO   LOW(C-H)    HIGH(C-H)   NO   LOW(C-H)    HIGH(C-H)
          0    17  2       18 14       1   519  0       519 13
```

*Figure 8   Extent consolidation: data set view*

After the allocation the space remains as one track primary and one track secondary. When the table is created, the space allocated jumps from one track to four tracks. No data at this point has been loaded. Tracks allocated increased as we started inserting data, but the number of extents remained at one. As more data was inserted and additional space was requested, the VTOC found free space adjacent to the tracks already allocated, so combined the extents into one extent. Extent consolidation occurred until the allocation for track 29. From extents 1 through 28 the table space occupied tracks starting at cylinder 17 head 2 (track 2 on cylinder 17) all the way through cylinder 18 head 14 (track 14 of cylinder 18).

Without extent consolidation, this initial 28-track range would have occupied 28 tracks with 28 physical extents instead of 28 tracks and one physical extent. When track 28 was full and another track was requested for track 29, cylinder 19 head 0 already had a data set occupying the location. There was no free space available until cylinder 519 head 0 (track 1). The table space's second physical extent started allocating space on cylinder 519 head 0 (track 1). After this point an additional 13 tracks were written. The second extent used free space from cylinder 519 head 0 (track 1) through cylinder 519 head 13 (track 14). Were this table space non-SMS or prior to z/OS 1.5, it would have allocated 42 tracks with 42 extents rather than 42 tracks and two extents.

When extent consolidation is in effect in z/OS 1.5, the secondary space allocation quantity can be smaller than specified or calculated by the sliding scale based on the physical extent number. PTF UQ89517 for APAR PQ88665 corrects this problem by using the logical extent number together with the sliding scale.

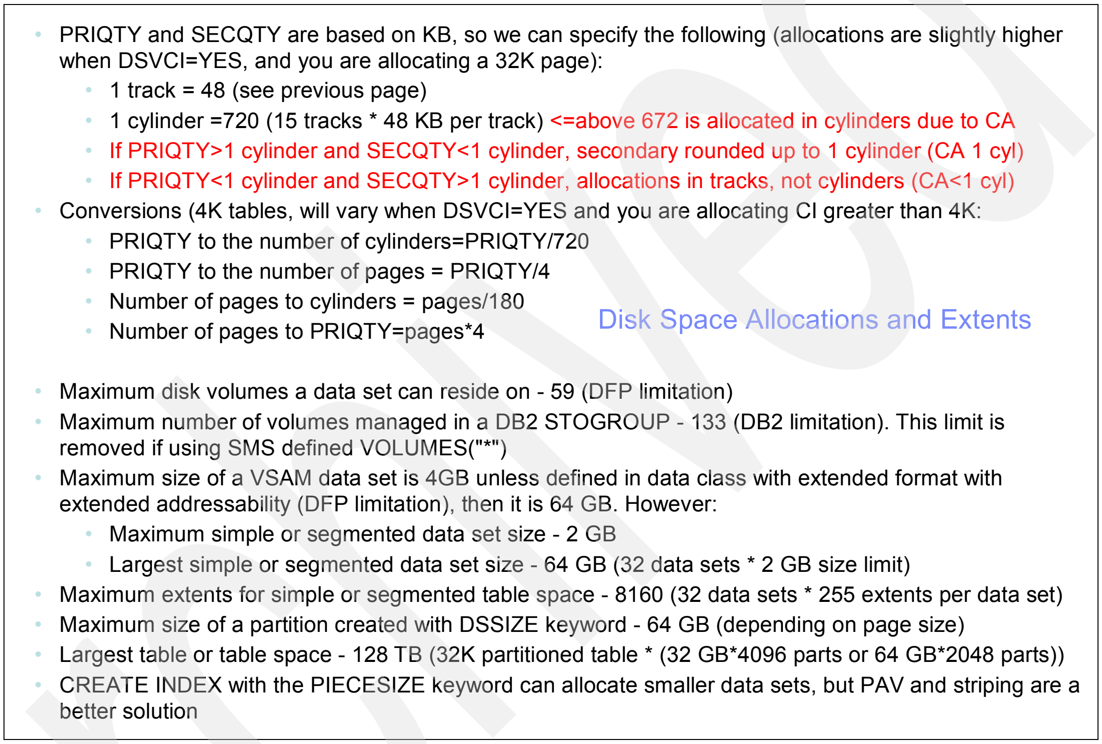
One cylinder is made up of 15 tracks. When you review the IEHLIST LISTVTOC information, keep in mind that head 0 actually refers to track 1 and head 14 is track 15.

## Disk space allocations and extents

Limitations for the allocation of DB2 and related data sets are a combination of DB2 and DFSMSdfp limitations.

Figure 9 shows that although the geometry of 3390s allow for allocation of 56 KB per track, having 4 KB, 8 KB, and 16 KB blocks means that we actually only use 48 KB per track of the 56 KB for our LDSs. Allocations are made on a multiple of 48 KB, meaning that allocating one track requires a quantity of 48. (Values are represented in kilobytes.)

A summary of limits on allocations and extents is provided in Appendix A, “Limits in DB2 UDB for z/OS” of the *DB2 UDB for z/OS Version 8 SQL Reference*, SC18-7426-03.



- PRIQTY and SECQTY are based on KB, so we can specify the following (allocations are slightly higher when DSVCI=YES, and you are allocating a 32K page):
  - 1 track = 48 (see previous page)
  - 1 cylinder =720 (15 tracks * 48 KB per track) <=above 672 is allocated in cylinders due to CA
  - If PRIQTY>1 cylinder and SECQTY<1 cylinder, secondary rounded up to 1 cylinder (CA 1 cyl)
  - If PRIQTY<1 cylinder and SECQTY>1 cylinder, allocations in tracks, not cylinders (CA<1 cyl)
- Conversions (4K tables, will vary when DSVCI=YES and you are allocating CI greater than 4K:
  - PRIQTY to the number of cylinders=PRIQTY/720
  - PRIQTY to the number of pages = PRIQTY/4
  - Number of pages to cylinders = pages/180
  - Number of pages to PRIQTY=pages*4

Disk Space Allocations and Extents

- Maximum disk volumes a data set can reside on - 59 (DFP limitation)
- Maximum number of volumes managed in a DB2 STOGROUP - 133 (DB2 limitation). This limit is removed if using SMS defined VOLUMES("*")
- Maximum size of a VSAM data set is 4GB unless defined in data class with extended format with extended addressability (DFP limitation), then it is 64 GB. However:
  - Maximum simple or segmented data set size - 2 GB
  - Largest simple or segmented data set size - 64 GB (32 data sets * 2 GB size limit)
- Maximum extents for simple or segmented table space - 8160 (32 data sets * 255 extents per data set)
- Maximum size of a partition created with DSSIZE keyword - 64 GB (depending on page size)
- Largest table or table space - 128 TB (32K partitioned table * (32 GB*4096 parts or 64 GB*2048 parts))
- CREATE INDEX with the PIECESIZE keyword can allocate smaller data sets, but PAV and striping are a better solution

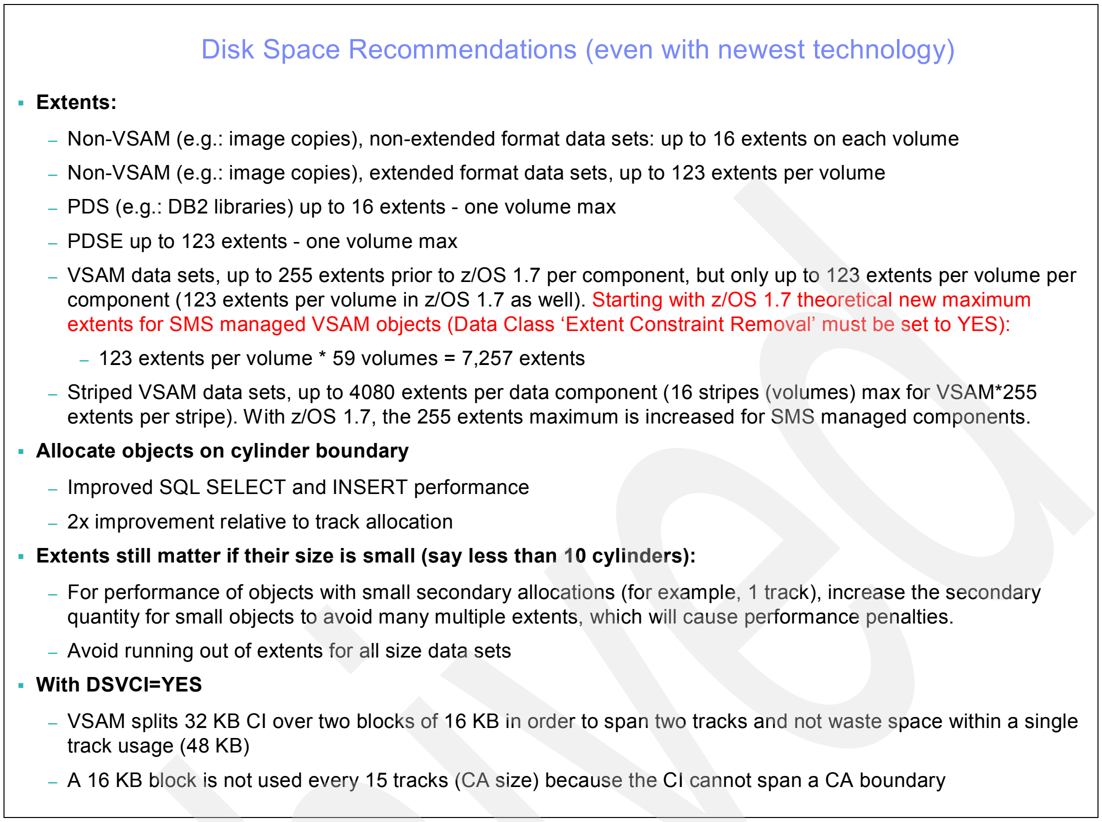
*Figure 9 Disk space allocations and extents*

Allocation of 32 KB table spaces results in additional 2.2% added to allocations (Figure 10 on page 16).

PRIQTY and SECQTY being below or above the equivalent of one cylinder determines the size of the control area (CA) size, so it is part of the equation for the number of CIs per CA.

Some of the limitations are DSMSdfp-based, and others are DB2-based. Therefore, it is important to be familiar with both the DB2 limitations as well as the DFSMdfp ones. You want to keep track of any changes to the limitations, whether associated with DB2 or DSMSdfp.

Partitions and LOBs may be created with the DSSIZE keyword, which allows for allocations greater than 2 GB in size up to a maximum of 64 GB, depending on the number of partitions created. The SMS Data Class attributes for both extended format (EF) and extended addressable (EA) must be on in order for partitions or LOBs to be created greater than 4 GB. These types of data sets are commonly called EF (extended format) or EA (extended addressable) data sets. DB2 managed STOGROUPs have a 133-volume limitation, but you may have more than 133 volumes in the SMS STOGROUP by using VOLUMES("*").



Disk Space Recommendations (even with newest technology)

- **Extents:**
  - Non-VSAM (e.g.: image copies), non-extended format data sets: up to 16 extents on each volume
  - Non-VSAM (e.g.: image copies), extended format data sets, up to 123 extents per volume
  - PDS (e.g.: DB2 libraries) up to 16 extents - one volume max
  - PDSE up to 123 extents - one volume max
  - VSAM data sets, up to 255 extents prior to z/OS 1.7 per component, but only up to 123 extents per volume per component (123 extents per volume in z/OS 1.7 as well). Starting with z/OS 1.7 theoretical new maximum extents for SMS managed VSAM objects (Data Class 'Extent Constraint Removal' must be set to YES):
    - 123 extents per volume * 59 volumes = 7,257 extents
  - Striped VSAM data sets, up to 4080 extents per data component (16 stripes (volumes) max for VSAM*255 extents per stripe). With z/OS 1.7, the 255 extents maximum is increased for SMS managed components.
- **Allocate objects on cylinder boundary**
  - Improved SQL SELECT and INSERT performance
  - 2x improvement relative to track allocation
- **Extents still matter if their size is small (say less than 10 cylinders):**
  - For performance of objects with small secondary allocations (for example, 1 track), increase the secondary quantity for small objects to avoid many multiple extents, which will cause performance penalties.
  - Avoid running out of extents for all size data sets
- **With DSVCI=YES**
  - VSAM splits 32 KB CI over two blocks of 16 KB in order to span two tracks and not waste space within a single track usage (48 KB)
  - A 16 KB block is not used every 15 tracks (CA size) because the CI cannot span a CA boundary

*Figure 10 Allocating space: recommendations*

Allocations of DB2 LDS data sets are always done using 48 KB of the 56 KB allowed for a 3390 track. This can be proved by using Table 3 and multiplying the VSAM physical block size by blocks per track. Whether 4*12 or 6*8 or 16*3, it all equates to 48 KB per track.

*Table 3 DB2 allocation sizes*

| DB2 page size | VSAM CI SIZE | | VSAM physical block size | | Blocks per track | | DB2 pages per track |
|---|---|---|---|---|---|---|---|
| | V7 | V8 | V7 | V8 | V7 | V8 | |
| 4 | 4 | 4 | 4 | 4 | 12 | 12 | 12 |
| 8 | 4 | 8 | 4 | 8 | 12 | 6 | 6 |
| 16 | 4 | 16 | 4 | 16 | 12 | 3 | 3 |
| 32 | 4 | 32 | 4 | 16 | 12 | 3 | 1.5 |

As seen in the table in Figure 11 on page 17, 32 KB tables allocate 1.5 pages per track. By doing so we add an additional 16 KB per track, thereby using the entire 48 KB of a track rather than just 66% of it had only 32 KB been allocated per track. Using this mechanism allows for three 32 KB pages (rather than just two) to be allocated on two tracks. Based on a one-cylinder CA, the result is an additional seven 32 KB pages per cylinder.

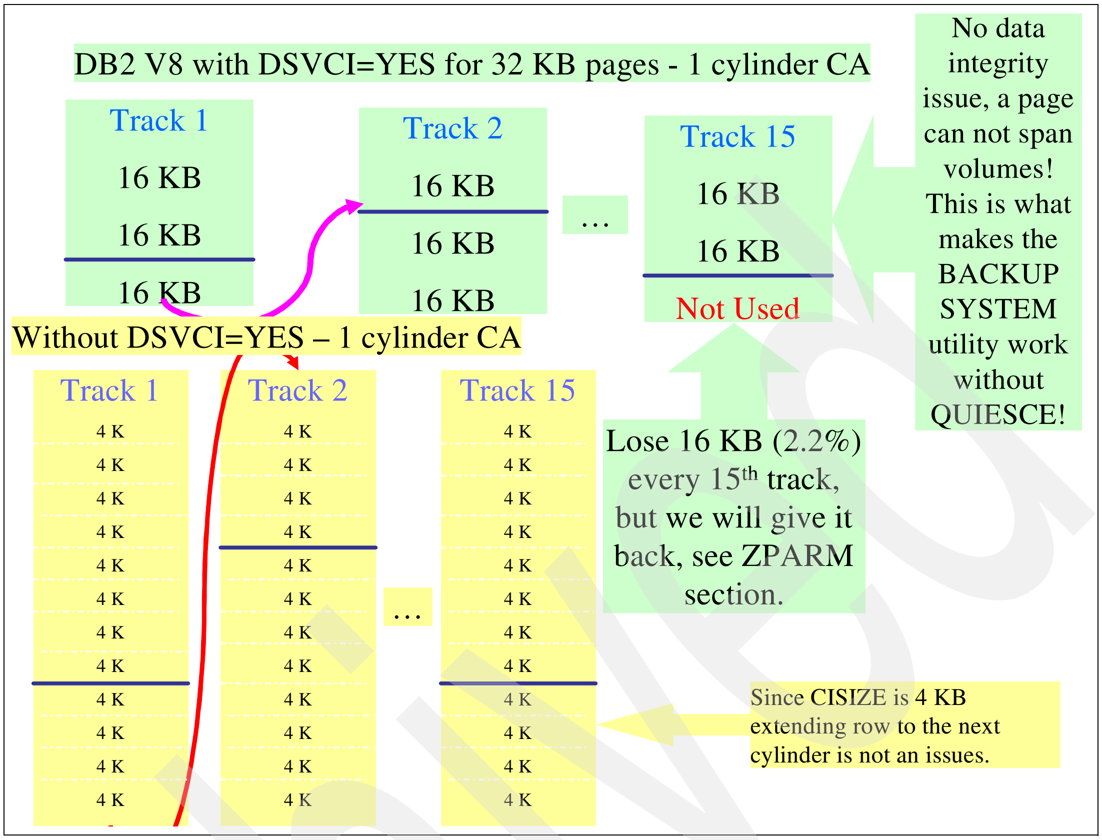

DB2 V8 with DSVCI=YES for 32 KB pages - 1 cylinder CA
Track 1
16 KB
16 KB
16 KB
Track 2
16 KB
16 KB
16 KB
…
Track 15
16 KB
16 KB
Not Used
No data integrity issue, a page can not span volumes! This is what makes the BACKUP SYSTEM utility work without QUIESCE!
Without DSVCI=YES – 1 cylinder CA
Track 1
4 K
Track 2
4 K
…
Track 15
4 K
Lose 16 KB (2.2%) every 15th track, but we will give it back, see ZPARM section.
Since CISIZE is 4 KB extending row to the next cylinder is not an issues.

*Figure 11 CI sizes with and without DSVCI=YES*

DB2 V8 introduced CI sizes that are the same size as the page being allocated. This is done through DSNZPARM DSVCI (discussed in more detail in “New CI sizes” on page 33). Making the CI size the same as the page size allows for data integrity because we can be sure that the whole row is intact when externalized. Prior to this enhancement there were some instances of data integrity exposures for 8 KB, 16 KB, and 32 KB objects when writing to disk at the extent boundary. The integrity exposure caused DB2 not to allow 8 KB, 16 KB, and 32 KB objects to use VSAM multi striping, and required a SET LOG SUSPEND command when doing split mirror design backups. Also, Concurrent Copy required SHRLEVEL REFERENCE for 32 KB pages and did not allow CHANGE. With DSVCI the integrity issue has been resolved and these functions are now available for all page sizes.

There are two major differences when allocating a 32 KB table space with DSVCI=YES:

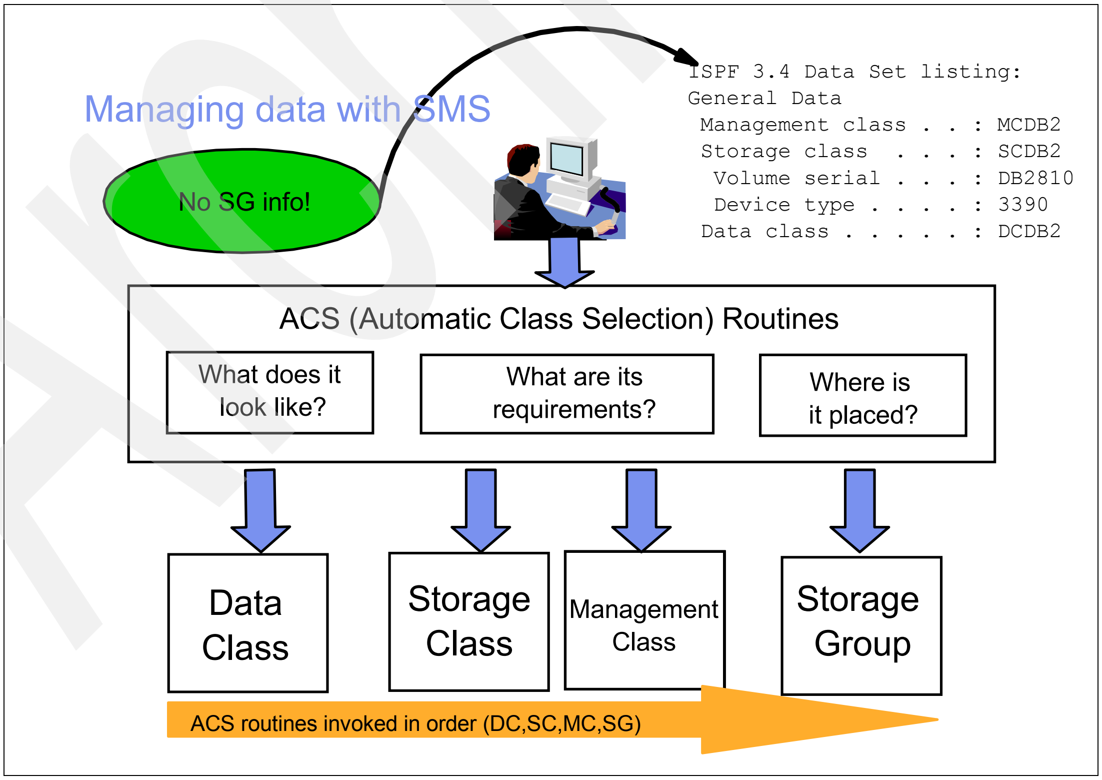
- Physical size of objects with DSVCI=YES write 16 KB physical blocks. Without DSVCI=YES, 4 KB physical blocks are written.
- For a 32 KB page table space with DSVCI=YES, the CI size is 32 KB; without DSVCI=YES, the CI size is 4 KB. Because a 32 KB table space with DSVCI=YES uses a 32 KB CI size, it is made up of two physical 16 KB pages, allowing for one 32 KB CI. With a 1-cylinder CA for 32 KB table spaces, because we write 1.5 pages per track, for the second page the CI includes the last 16 KB of the first track, and the next track contains the remaining 16 KB. This is because of the rules of DSMSdfp, in which a CI cannot expand beyond the size of a CA. With DSVCI=YES, if the 32 KB page table space was created with a 1-cylinder CA, the last 16 KB of the cylinder are not used, resulting in the loss of 2.2% of a cylinder. The reason for this is because when DB2 tries to allocate the last 16 KB of data and extend to a new cylinder to write the second block for the rest of the CI, the CI would span from one CA to another, which is a DSMSdfp violation. Without

DSVCI=YES, all DSMSdfp knows is that we are writing out twelve 4 KB records per track with a CI size of 4 KB. DSMSdfp is unaware that we are requesting a 32 KB page, and instead DB2 strings together eight 4 KB pages and returns to us an image of a 32 KB page. This is not the case with DSVCI=YES where the CI size of a 32 KB table space is 32 KB, therefore there is no need to string data together in such a fashion. From an integrity point of view we need to look at how allocations work without DSVCI=YES. The last 16 KB of a cylinder is used when you have a 1-cylinder CA, so there is no loss of 2.2% per cylinder. At the end of cylinder 1 where only four of the 4 KB pages have been written, a new cylinder is required to complete the other half of the page, the remaining four 4 KB pages. The issue here is when cylinder 1 is at end of a volume and cylinder 2 gets allocated to a new volume. In this case, the first half of one 32 KB page resides on one volume, and the second half of the page resides on a totally different volume. Based on write and backup mechanisms, DB2 cannot guarantee that the entire 32 KB page will always be consistent, hence such things as VSAM multi striping and concurrent copy require a SET LOG SUSPEND command and are not supported for objects above 4 KB without DSVCI=YES.

For more information about not being able to stripe above a 4 KB page without DSVCI=YES, before DB2 V8, see APAR PQ53571.

## SMS

When we deal with a Storage Administrator in regard to disk, it typically has to do with requesting additional volumes. The Storage Administrator has many options they can choose to set up and work with the storage environment. In this section we discuss some of the options that DB2 can interface with (Figure 12).



```
ISPF 3.4 Data Set listing:
General Data
 Management class . . : MCDB2
 Storage class   . . . : SCDB2
  Volume serial . . . : DB2810
  Device type . . . . : 3390
 Data class . . . . . : DCDB2
```

*Figure 12   SMS*

Storage Administrators code ACS (Automatic Class Selection) routines combined with settings in ISMF to provide an SMS environment. Allocations and rules for your SMS managed data set are provided in the ACS routines.

After the allocation is complete, the name of the Data Class, Storage Class, and Management Class are provided as part of LISTCAT output as well as ISPF data set space information, but at no time is the Storage Group name provided. The Storage Group name is provided under certain circumstances, such as in some error messages.

## Data set separation

Data set separation enables you to designate groups of data sets in which all SMS-managed data sets in a group are kept separate, on the physical control unit (PCU) level, and from all the other data sets in the same group. This reduces the effects of single points of failure.

Using this technique is a very good way of allocating the active log and BSDS data sets for availability purposes. If one physical disk controller is no longer available, the active log and bootstrap data sets should be available on another physical disk controller.

The separation data set cannot be used with non-SMS-managed data sets or with full volume copy utilities such as peer-to-peer remote copy (PPRC). Some alternatives are:

- Create the data set before PPRC is established.
- Break the pair after it is established, create the data set, then re-establish the pair.

Although we can physically separate data sets at allocation time, the Storage Administrator can move a separated volume from one disk controller right next to a volume you are trying to keep it separate from.

## Data class information

SMS is used for allocation of new data sets, and at times when data sets are recalled from DFSMShsm™. Data Class allows for a collection of attributes to be specified for a grouping of data sets. Attributes include such items as JCL data control block (DCB) information, space requirements, volume requirements, and even VSAM attributes for VSAM data sets.

For Data Class attributes, explicit specifications by end users in JCL override any parameters derived from the data class ACS routine. If SMS is active, a data class can be assigned to any data set. For non-SMS-managed disk data sets, the system uses the allocation attribute values of the data class, but it does not save the data class name. For tape data sets, only the expiration and retention values are applied. Not all Data Class attributes can be used for non-SMS managed data sets.

Data Class attributes are very useful for installations that use the same DCB and same attributes for a large number of data sets. Let us look at an example where you have 100 data sets that have DCB attributes of:

```
RECFM=FB,LRECL=80,BLKSIZE=8000,SPACE=(CYL,(2,1))
```

Instead of coding all of the parameters for every DD statement, all that has to be on the DD statement is the data set name, the disposition of NEW, and one of the following options:

- The Storage Administrator can set up the Data Class to automatically provide the attributes either with a pattern or with specific data set names.
- The Storage Administrator allows users to add a DATACLAS parameter of their DD to explicitly assign a Data Class.

Extended format (EF) and extended addressable (EA) are assigned by the Data Class. Both EF and EA are required when allocating DB2 LDSs greater than 4 GB in size for partitions or LOBs. EF is required for VSAM data striping as well. EF/EA recommendations:

- For LDSs and FICON only, there can be significant performance issues when running at high path utilization. Resolution for this performance issue comes in the form of a modified channel control word (CCW) called modified indirect address word (MIDAW). In order to use MIDAW, you must have a z9 processor and have z/OS 1.7 or 1.6 plus PTFs. This recommendation pertains only to FICON and does not include ESCON. Do not allow the Data Class to assign EF/EA to LDSs when using FICON unless needed for space requirements (going above the 4 GB limit) until MIDAW has been implemented.

  **Note:** For more information regarding the performance issue and resolution, refer to "The MIDAW facility: improved channel architecture" on page 38. MIDAW improves the performance of small block sizes.

- Sequential data sets do not suffer from the same performance problems as LDSs with EF/EA assigned; even when using FICON, they generally use large blocks of data. Performance results were better or at par when assigning EA/EF to sequential data sets. Some processing such as BDAM is not permitted, which means that the DB2 archive log data sets cannot be assigned EF/EA unless you purchase a special product to allow for EF/EA, such as the IBM DB2 Archive Log Accelerator for z/OS. EF/EA also benefits sequential data sets by increasing the number of extents allowed for data sets from 16 per volume to 123 per volume. Image copy data sets are a good candidate to manage using EF/EA.

  The Data Class can also assign an attribute that would help resolve some space-related problems. During allocation, there might not be enough space on volumes to meet the requested space. SMS volume selection can sometimes solve this problem by trying all candidate volumes before failing the allocation. You can also use the Space Constraint Relief and Reduce Space Up To (%) attributes to request that an allocation be retried with a space request reduced by a certain percent if it fails due to space constraints. SMS retries the allocation by combining any of the following tasks:

  – Spreading the requested quantity over multiple volumes. This option is not permitted for data sets allocated with the guaranteed space attribute for DB2 managed data sets.

  – Allocating a percentage of the requested quantity. This option is not permitted for data sets allocated with the guaranteed space attribute.

  – Using more than five extents for primary allocations for a new volume or another volume that is being used for additional extents.

VSAM and non-VSAM multi-striped data sets do not support space constraint relief. However, single-striped VSAM and non-VSAM data sets can use space constraint relief.

Certain attributes such as DYNAMIC VOLUME count should *not* be used for DB2 LDSs. You might want to use such things as VOLUME COUNT for your image copy data sets where an additional number of volumes can be requested.

z/OS 1.7 allows for VSAM LDSs to grow to a theoretical limit of 7,257 extents. Your Storage Administrator must set Extent Constraint Removal to YES in order to exceed the previous limit of 255 extents per component.

## Storage class

Without SMS, you must place critical data sets on selected storage devices in order to improve performance.

A storage class is a list of storage objectives and requirements. Each storage class represents a list of services that are available to data sets and objects having similar access requirements. A storage class does not represent any physical storage, but rather provides the criteria that SMS uses in determining an appropriate location to place a data set.

Storage Class is where SMS determines whether an allocation should continue down the path after completion of the Storage Class to the Management Class and finally the Storage Group, or if at this point a data set should be non-SMS managed and not continue on.

The major characteristics of the Storage Class are defined as follows:

- Defining Performance Objectives

  Much of the MSR (millisecond response time) related values for disks (for instance, if cache is used) are basically ignored with newer disks. Some values are still used, such as the combination of MSR rate Sustained Data Rate (SDR) and Data Class, to determine the number of stripes for data sets.

- Defining Availability

  Used to determine whether the data set is eligible for RAID or dual copy volumes (or both).

- Defining Accessibility

  Used to determine requirements for point-in-time copy, using either concurrent copy, virtual concurrent copy, or FlashCopy (for ESS and DS8000).

- Defining the Guaranteed Space Attribute

  – Guaranteed space honors your volume request, and although it is allowed for DB2 data sets, it is not recommended. For example, you have a pool of 100 disks for your DB2 data sets, with one of the volumes being DB200A. If guaranteed space is allowed and you request allocation to volume DB200A, then the data set is allocated only to DB200A and none of the other 99 volumes is eligible. If guaranteed space is not used and you try and allocate to volume DB200A, SMS allocates your data set to one of the 100 volumes, not necessarily DB200A. If DB200A is picked, it was the best choice of volumes from an SMS preference value perspective, not because you requested DB200A. Another major drawback is when a data set goes multi-volume. In this case the primary and not the secondary allocations are used on the next volume, which can be much a much larger space requirement than many a customer disk can handle. This is not done for DB2 managed data sets where the secondary allocation is used when going multi-volume even if guaranteed space is specified. User-managed DB2 data sets require the primary allocation on all volumes when allocating a data set using guaranteed space.

  – There are a number of issues when using guaranteed space and therefore its use is discouraged. Guaranteed space is typically not needed with modern disk because of the use of PAV and MA. This is not to say that hot spots no longer exist on modern disk, but they are greatly reduced and often can be resolved by the Storage Administrator. They can still be used for allocating the BSDS and DB2 active log data sets, but the separation profile in SMS can accomplish this task as well.

Storage Class determines whether a multi-tiered Storage Group is used. Requesting or requiring PAVs is done through the Storage Class as well.

While reviewing the Storage Class with a Storage Administrator, take note of what type of data sets are allocated to what type of disk. You want your BSDS and active log data sets on your fastest disk. If for example you have a mix of DS8000 and ESS, you will want your BSDS and DB2 active log data sets on the newer DS8000 and your user data on the ESS. Begin moving important production data onto the DS8000 as time and space permit.

## Management class

A management class is a list of data set migration, backup, and retention attribute values. Management class also includes object expiration criteria, object backup requirements, and class transition criteria for management of objects. DFSMShsm uses the attributes of the management class that is associated with a data set to manage storage. When you assign a management class to a system-managed disk data set, SMS places the management class name in both the basic catalog structure (BCS) and the VSAM volume data sets (VVDS) catalog entries of the data set. Management class is optional for system-managed data sets and does not apply to data sets that are not system managed. Management class is not used for tape data sets. Managed functions include:

- Requirements for releasing over-allocated space
- Migration requirements
- Retention criteria
- Treatment of expired data sets
- Frequency of backup
- Number of backup versions
- Retention of backup versions
- Number versions
- Retain only version
- Retain only version unit
- Retain extra versions
- Retain extra versions unit
- Copy serialization
- Generation data group (GDG) information

GDG management is simplified with Management Class. For example, you can request SMS to keep only the current version (0) of a GDG data set on disk and migrate all other versions. You may also want a convention where versions 0 and -1 are kept, but all others are migrated. The Management Class is flexible enough to handle these types of requests.

Some functions should be discussed with your Storage Administrator, such as the Partial Release Attribute on the ISMF panel for Management Class. This option tells DFSMShsm that it may compress any created VSAM data sets in extended format (EF) and is not using the guaranteed space attribute. You do not want this type of function turned on for your DB2 LDSs. Allowing this option can reduce the space allocated for large LDSs, which, if not read only or read mostly can cause you to request more space and extents than originally required.

Expiration of data is a function for some DB2 subsystem parameters (DSNZPARMs) and activities that you can specify, but it does not request z/OS to complete the task. For example, DSNZPARM ARCRETN tells DB2 how long to retain the DB2 archive log data sets; however, when this date meets the criteria, a request is not made to z/OS to expire the data set. One way of accomplishing this type of task is by using the Management Class to expire the DB2 archive log with a time frame that matches your DSNZPARM value. DFSMSrmm™ (RMM, which stands for removable media manager) can also be used to expire the DB2 archive log data sets if they reside on RMM-managed tape.

Other functions in DB2 utilities fall into the same category. Executing the MODIFY DELETE utility for image copy data sets modifies the information in the DB2 catalog and directory, but when an image copy is deleted DB2 does not notify z/OS that a delete or expire should be executed for the data set. Such a case requires the same type of methodology used for the DB2 archive log data sets. Either a Management Class is required to expire the image copy data set after the number of days specified in the MODIFY DELETE utility or request RMM to expire the data set if managed by RMM.

## DFSMShsm space management

As shown in Figure 13, migration has three categories:

- Level 0

  This is where your data, such as your DB2 LDSs reside. Other disk data sets that have not been migrated are considered level 0 data sets as well.

- ML1 (migration level 1)

  HSM-owned disk. If you migrate a data set you have a choice to migrate it from level 0 to ML1 or ML2. Recalling a migrated data set from ML1 to level 0 is typically fast because DFSMShsm is making a disk-to-disk move.

- ML2 (migration level 2)

  HSM-owned, typically tape. Data sets can be migrated from level 0 to ML1 or directly to ML2. Recalling data from ML2 tape might result in elongated wait times, especially if several data sets are being recalled from the same tape or your installation uses stand-alone non-automated tape drives.

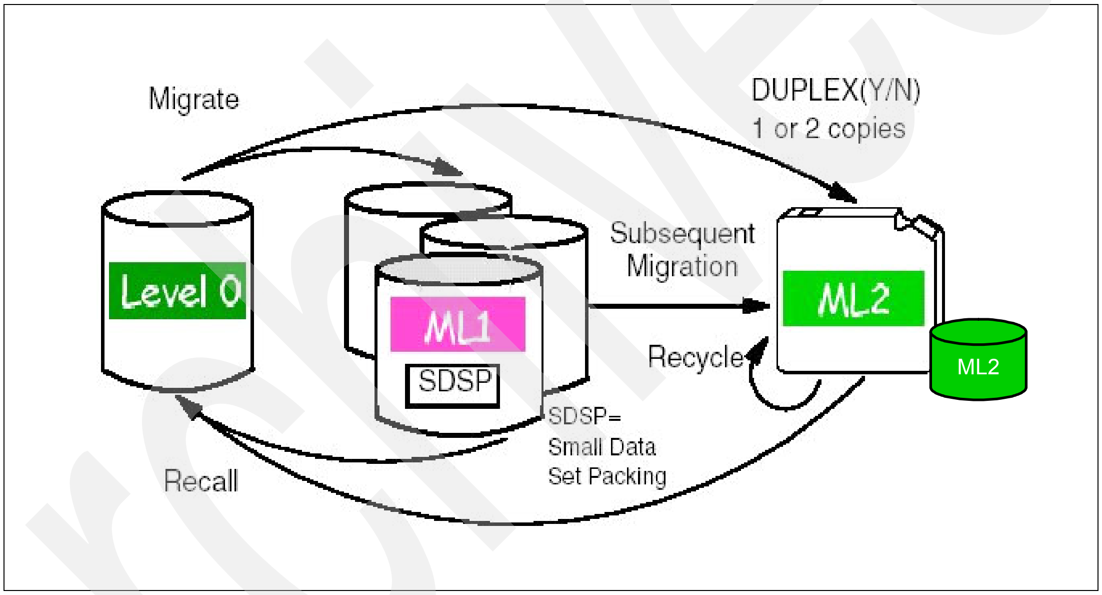


*Figure 13 DFSMShsm migration overview*

Data set migration can happen from level 0 to ML1 and then to ML2 or directly to ML2. For example, you can have DFSMShsm migrate your data set from Level 0 to ML1 if not used in 100 days. If the data set has not been used for an additional 50 days, have DFSMShsm migrate the data set from ML1 to ML2. In this case the data set would migrate to ML2 only if not used in 150 days. Or you might have data for which the chance of requiring a DFSMShsm RECALL of data are small, so an alternative to this would be to migrate directly from level 0 to ML2 tape. Migrating directly from level 0 to ML2 reduces the amount of ML1 disk required.

DFSMShsm can expire (delete) data sets on level 0, ML1, or ML2. For example, if a data set is not used on level 0 in 100 days, just expire the data set and do not migrate it. Data sets can also be expired from ML1 or ML2 if not used within the expiration time frame. Data sets without expiration dates on level 0, ML1, or ML2 never expire.

## SMS storage group

Storage groups represent the physical storage managed by SMS. This storage can be collections of disk volumes, volumes in tape libraries, volumes in optical libraries, or virtual input/output (VIO) storage. A storage group, used with storage classes, separates the logical requirements of accessing data from the physical requirements to store the data. You can use storage group attributes to specify how the system should manage the storage group. You use the storage group ACS routine to assign a new data set or object to a storage group. You can assign multiple candidate storage groups (except for objects), in which case the system chooses a specific storage group from your list. Storage group definitions are not apparent to users and are not externalized except during certain error conditions.

Decisions about whether to use DFSMShsm, DFSMSdss, or both functions (such as backup or migration) are made for all eligible volumes in the entire storage group.

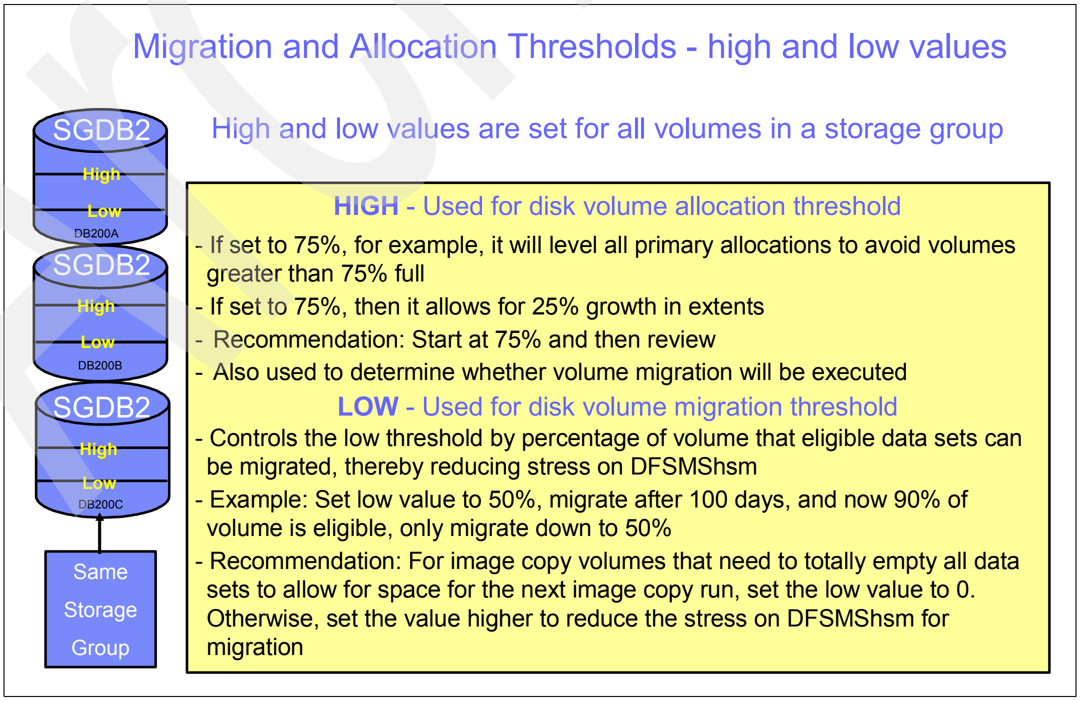
For installations that allow for DFSMShsm to migrate DB2 LDS data sets, verify that DSNZPARM RECALL is set to YES, and determine what amount of time should be set for DSNZPARM RECALLD. Some things to consider if your installation migrates your DB2 LDSs:

- How many objects have to be recalled at the same time?
- How many objects reside on the same DFSMShsm ML2 tape?
- Will serial recalls complete in a timely manner?

It may be possible to avoid most DB2 timeouts for the recall of migrated data sets by working with your Storage Administrator on a migration and recall strategy for DB2 LDSs.

The threshold value HIGH refers to the percentage of space on the disk that is available when you are allocating a data set's primary space allocation. SMS avoids that volume for allocations if it is above the specified threshold. The remaining portion of the disk is used to allow for data sets to expand in size on the disk and take additional extents if required.

Figure 14 shows an example with three volumes in Storage Group SGDB2: DB200A, DB200B, and DB200C.



Migration and Allocation Thresholds - high and low values

High and low values are set for all volumes in a storage group

**HIGH** - Used for disk volume allocation threshold

- If set to 75%, for example, it will level all primary allocations to avoid volumes greater than 75% full
- If set to 75%, then it allows for 25% growth in extents
- Recommendation: Start at 75% and then review
- Also used to determine whether volume migration will be executed

**LOW** - Used for disk volume migration threshold

- Controls the low threshold by percentage of volume that eligible data sets can be migrated, thereby reducing stress on DFSMShsm
- Example: Set low value to 50%, migrate after 100 days, and now 90% of volume is eligible, only migrate down to 50%
- Recommendation: For image copy volumes that need to totally empty all data sets to allow for space for the next image copy run, set the low value to 0. Otherwise, set the value higher to reduce the stress on DFSMShsm for migration

*Figure 14 SMS Storage Group high and low values*

If HIGH is set to 75, then when the volume exceeds the 75% threshold it will no longer accept requests for new data sets. It is still possible to allocate to this volume under certain circumstances, such as if all volumes are above the threshold and there is nowhere else to place the new data set. For example, if DB200A is 76% full, and volumes DB200B and DB200C are less than 75% full, a data set allocation typically will not use DB200A but will allocate the data set to DB200B or DB200C.

In addition, the HIGH threshold value is used by DFSMShsm to determine whether data sets should be migrated off a disk volume in the storage group.

The LOW threshold value is used as the threshold goal in reducing the amount of space occupied on a volume in the storage group during DFSMShsm interval migration or daily space management.

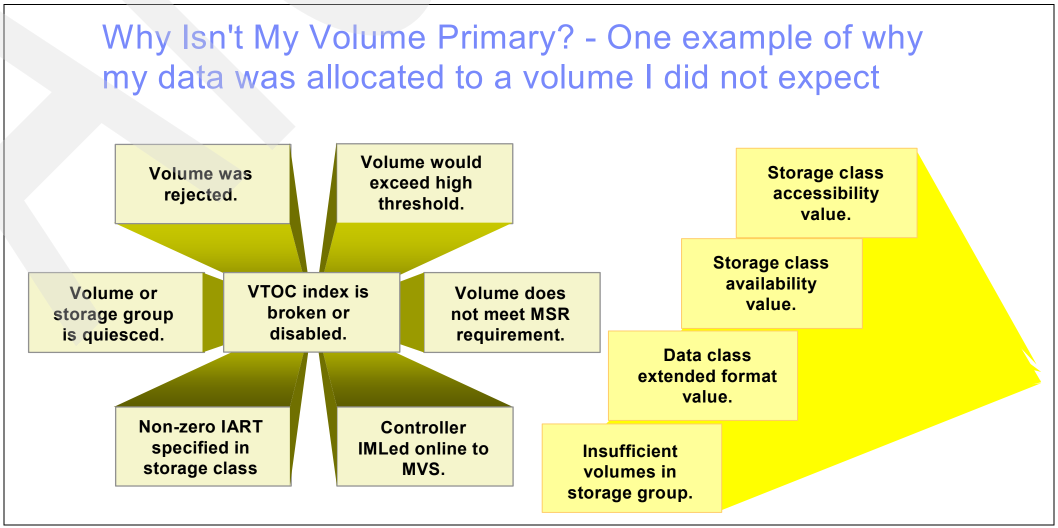
Discuss the HIGH setting with your Storage Administrator. Some installations set the value to 95% or 99%, but this can cause many multi-volume data sets. As a general rule, start at 75% and decide in time whether HIGH should be a different number.

Besides the normal Storage Group used for your DB2 LDSs, your Storage Administrator might assign additional Storage Groups as added insurance so that a data set allocation or extent request will succeed. Storage Administrators may use a mix of your Storage Group and additional volumes from an extend or overflow Storage Group. If SMS determines that space cannot be allocated to the Storage Group normally assigned for your DB2 LDSs, it may choose a volume other than what you expect in order to ensure that your allocation does not fail. If this is the case, you *must* be sure that all volumes that can be used for allocation are included in your volume copies, such as full volume dumps or FlashCopy operations.

The following items result from defining a multi-tiered Storage Group in the Storage Class with `SET &STORGRP = 'SG1', 'SG2', 'SG3'`:

- SMS selects candidate volumes from SG1 before SG2 and SG3.
- If all enabled volumes in SG1 are over threshold, then SMS selects from SG2.
- If all enabled volumes in SG2 are over threshold, then SMS selects from SG3.
- If all volumes are over the threshold, then SMS selects from the quiesced volumes in the same order.

Figure 15 shows reasons why a volume might not be considered for primary allocation.



Why Isn't My Volume Primary? - One example of why my data was allocated to a volume I did not expect

Volume was rejected.

Volume would exceed high threshold.

Volume or storage group is quiesced.

VTOC index is broken or disabled.

Volume does not meet MSR requirement.

Non-zero IART specified in storage class

Controller IMLed online to MVS.

Storage class accessibility value.

Storage class availability value.

Data class extended format value.

Insufficient volumes in storage group.

*Figure 15   Some reasons for primary allocation failures*

SMS can be set up to allow for volumes to be selected as primary, secondary, tertiary, or rejected. Weighing factors for each pass determine where and whether a data set will be allocated. Data Class and Storage Class information, as well as other information, is passed to the Storage Group that determines allocation.

## Copypools

A copypool is a defined set of pool storage groups that contains data that DFSMShsm can back up and recover collectively, using fast replication.

DFSMShsm manages the use of volume-level fast replication functions, such as FlashCopy and SnapShot. These functions provide point-in-time copy services that can quickly copy data from a source location to a target location. Using a copypool, you can specify the pool storage groups that you want DFSMShsm to process collectively for fast replication.

DB2 BACKUP and RESTORE utilities invoke DFSMShsm in order to execute a copy or a restore of the copypools containing DB2 data or logs. Control records are kept in the BSDS and in the DFSMShsm backup control data set (BCDS).

DSN$*locn-name*$*cp-type*, DSN, and $ are required. *locn-name* is the DB2 location name, and *cp-type* is the copypool type. DB is for database, and LG is for logs.

For example, in Figure 16, DB2 DB1P would have copypools named DSN$DB1P$DB (locn-name is DBP1P) for the database copypool (cp-type DB) and DSN$DB1P$LG for the log copypool (cp-type LG).

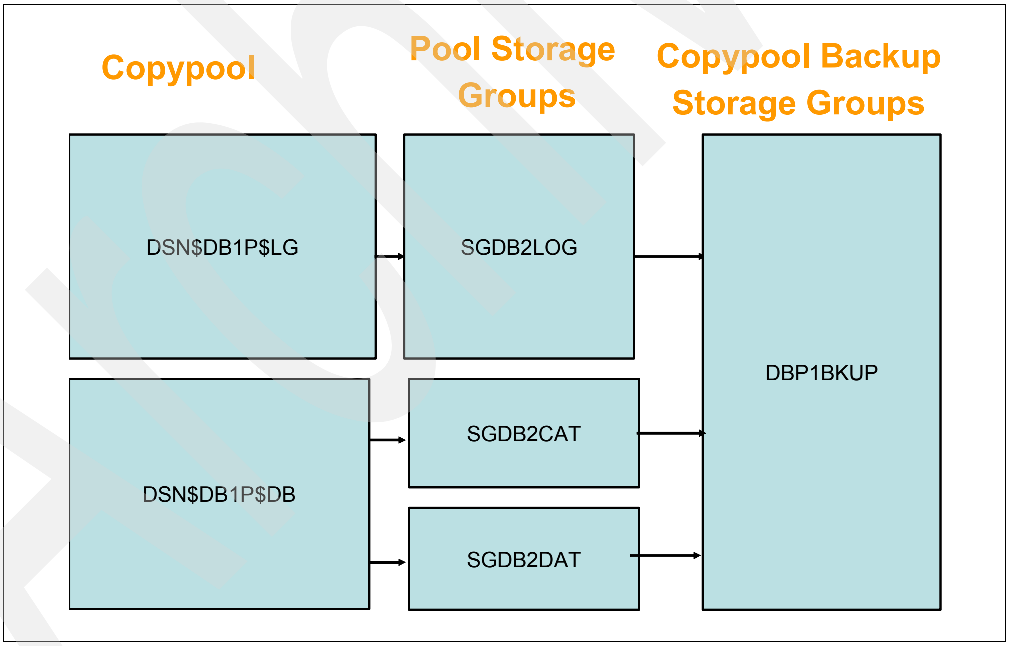


*Figure 16   Copypools*

# DSNZPARMs relating to disk storage management

Several different DSNZPARMs relate to storage functions. In this section we discuss what DSNZPARMs work best with typical storage environments.

## Sliding secondary allocation size

DB2 V8 adds a new function called *sliding secondary space allocation*, which can be turned on automatically in DB2 V8 and is used for DB2 managed data sets (STOGROUP defined).

The enhancement helps to reduce the number of out-of-space conditions, eliminate need for PRIQTY and SECQTY specification on SQL CREATE, improve user productivity, and avoid the performance penalty associated with small extent sizes.

This patented solution benefits all users of DB2. It delivers autonomic selection of data set extent sizes with a goal of preventing out-of-extent errors before reaching maximum data set size. It particularly benefits users of ERP/CRM vendor applications, which have many small data sets that can grow rapidly.

The V8 new DSNZPARM parameter is MGEXTSZ (with a global scope), option 7 on install panel DSNTIP7. It determines what happens if a DB2 LDS is out of space and DB2 needs to create a new data set (piece). The values are YES and NO. The default value for MGEXTSZ is NO. Sliding secondary is always on, whether you specify YES or NO. The options have the following effects:

- NO

  When specifying a value for PRIQTY and SECQTY, the new data set (created after A001) will have the same allocation as A001 for primary and secondary and therefore take on the characteristics of A001's PRIQTY and SECQTY. If PRIQTY and SECQTY were not specified then the allocation works as documented in YES.

- YES

  When allocating a new data set for a new piece (after A001), the primary and secondary allocation will be the last size calculated by the sliding secondary of the previous piece. The maximum allocation will not exceed 127 cylinders for objects 16 GB or below, and 559 cylinders for objects 32 GB or 64 GB.

  For example, A001 reaches the 2 GB limit and DB2 now needs to create the A002 data set. If A001's last allocation based on sliding scale was 79 cylinders, A002 will be created with a primary and secondary allocation of 79 cylinders.

  The maximum allocation will not exceed 127 cylinders for objects 16 GB or below, and 559 cylinders for objects 32 or 64 GB.

Figure 17 on page 28 shows the sliding scale mechanism for 64 GB data sets with 255 extents. It assumes an initial extent of one cylinder.

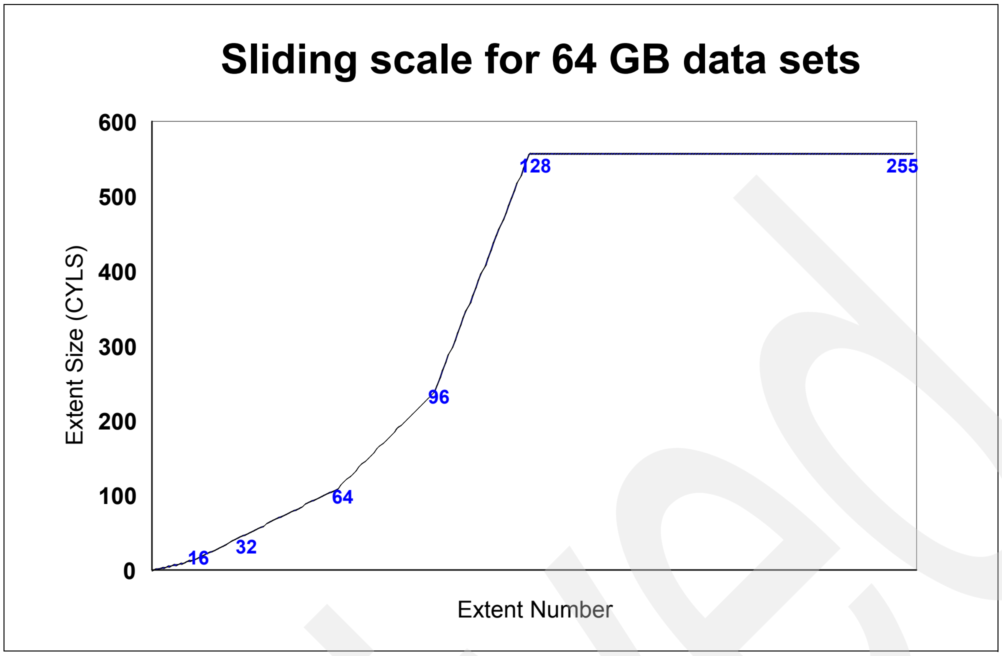


*Figure 17 Sliding scale for 64 GB data sets*

Figure 18 shows the sliding scale mechanism for a 16 GB data set with a maximum of 255 extents. It assumes an initial extent of 1 cylinder.

An increasing secondary quantity size up to 127 extents is allocated and a constant number of 127 cylinders for data sets less than 1 GB or 1, 2, 4, 8 and 16 GB is used for the secondary allocation thereafter. This approach of sliding the secondary quantity minimizes the potential for wasted space by increasing the extents size slowly at first, and it avoids very large secondary allocations from extents 128-255, which most likely cause fragmentation where multiple extents have to be used to satisfy a data set extension. The solution addresses newly allocated data sets as well as existing data sets requiring additional extents.

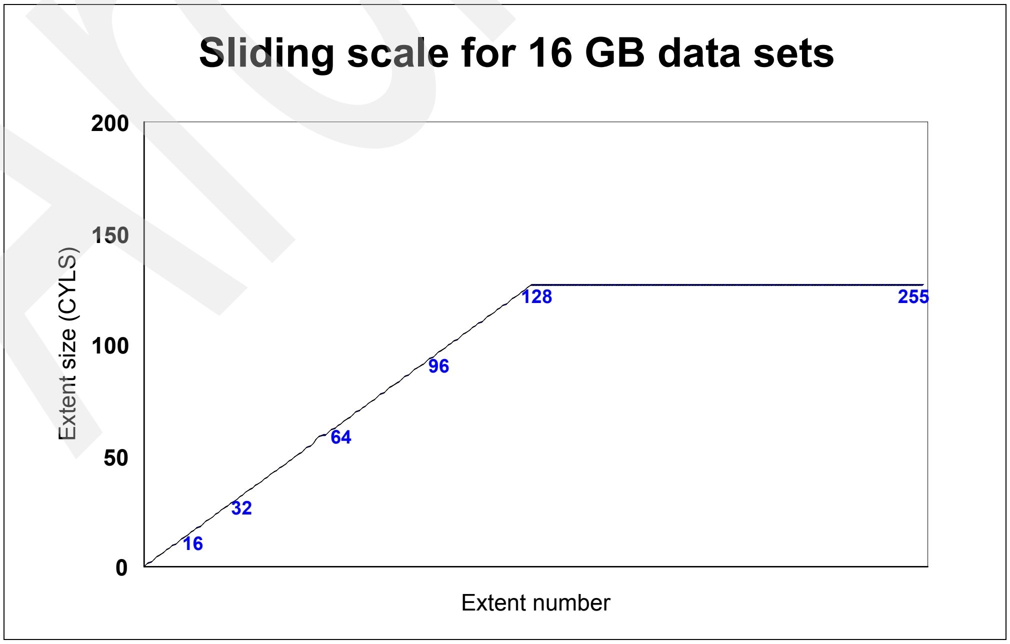


*Figure 18 Sliding scale for 16 GB data set*

With z/OS 1.7, for SMS managed data sets with the Extent Constraint Removal option in the SMS data class set to YES, the theoretical extent limit has been increased to 7,257. (This assumes a data set allocation over the maximum of 59 volumes, each volume with 123 extents.) The PTFs for APARs PK07590 and PK10594 for DB2 V8 and V7, as well as the PTF for OA12434 for VSAM, are needed for this function.

DB2 can apply the sliding secondary beyond the 255 extent limit if necessary. Requiring more than 255 extents is typically the result of heavy volume fragmentation. When discussing extents, keep in mind that if extent consolidation is used, in this case DB2 still provides the correct calculation based on size.

The value for PRIQTY plays a major role in the sliding allocations. The actual value applied by DB2 for default PRIQTY is determined by the applicable TSQTY or IXQTY DSNZPARMs, which were introduced with DB2 V7 as option 4 and 5 on install panel DSNTIP7. DSNZPARMs TSQTY and IXQTY now have global scope. TSQTY applies to non-LOB table spaces. For LOB table spaces, a 10-times multiplier is applied to TSQTY to provide the default value for PRIQTY. IXQTY applies to indexes. DSNZPARMs TSQTY and IXQTY continue to have a default value of 0 (zero), but this value indicates that a new default value of 720 KB (1 cylinder) is to be applied. If TSQTY is set to 0, then one cylinder is the default PRIQTY space allocation for non-LOB table spaces and 10 cylinders is the default PRIQTY space allocation for LOB table spaces. If IXQTY is set to 0, then one cylinder is the default PRIQTY space allocation for indexes.

DB2 applies the following four rules when calculating the allocations (where the first three apply also to DB2 versions prior to V8):

- If PRIQTY is specified by the user, the PRIQTY value will be honored. Otherwise, if not, the new default value as determined by either TSQTY, TSQTY*10, or IXQTY will be applied: one cylinder for non-LOB table spaces and indexes, and 10 cylinders for LOB table spaces.
- If no SECQTY is specified by the user, the actual secondary quantity allocation will be determined by the maximum of 10% of PRIQTY, and the minimum of calculated secondary allocation quantity size using the slide scale methodology and 559 (or 127) cylinders depending on maximum DB2 data set size. When a page set spills onto a secondary data set, the actual secondary allocation quantity will be determined and applied to the primary allocation. The progression will then continue. Prior to DB2 Version 8, the PRIQTY would have been used.
- If SECQTY is specified by the user as 0 to indicate “Do not extend,” this will always be honored. The condition will apply to DSNDB07 work files where many users set SECQTY to zero to prevent work files growing out of proportion.
- If SECQTY is specified by the user as greater than 0 (and MGEXTSZ is set to YES), the actual secondary allocation quantity will be the maximum of the minimum of calculated secondary allocation quantity size using the slide scale methodology and 559 (or 127) cylinders depending on maximum DB2 data set size, and the SECQTY value specified by the user. When a page set spills onto a secondary data set, the actual secondary allocation quantity will be determined and applied as the primary allocation. The progression will then continue. Prior to DB2 Version 8, the PRIQTY would have been used.

The user can provide override values for TSQTY and IXQTY DSNZPARMs to avoid wasting excessive disk space. For example, on a development subsystem, TSQTY and IXQTY may be set to 48 KB for track allocation. The use of the default for PRIQTY is recorded in the associated PQTY column as -1 in the SYSTABLEPART or SYSINDEXPART catalog table.

DB2 always honors the PRIQTY value specified for the first piece (A001) by the user and recorded in the associated PQTY column in the SYSTABLEPART or SYSINDEXPART catalog table.

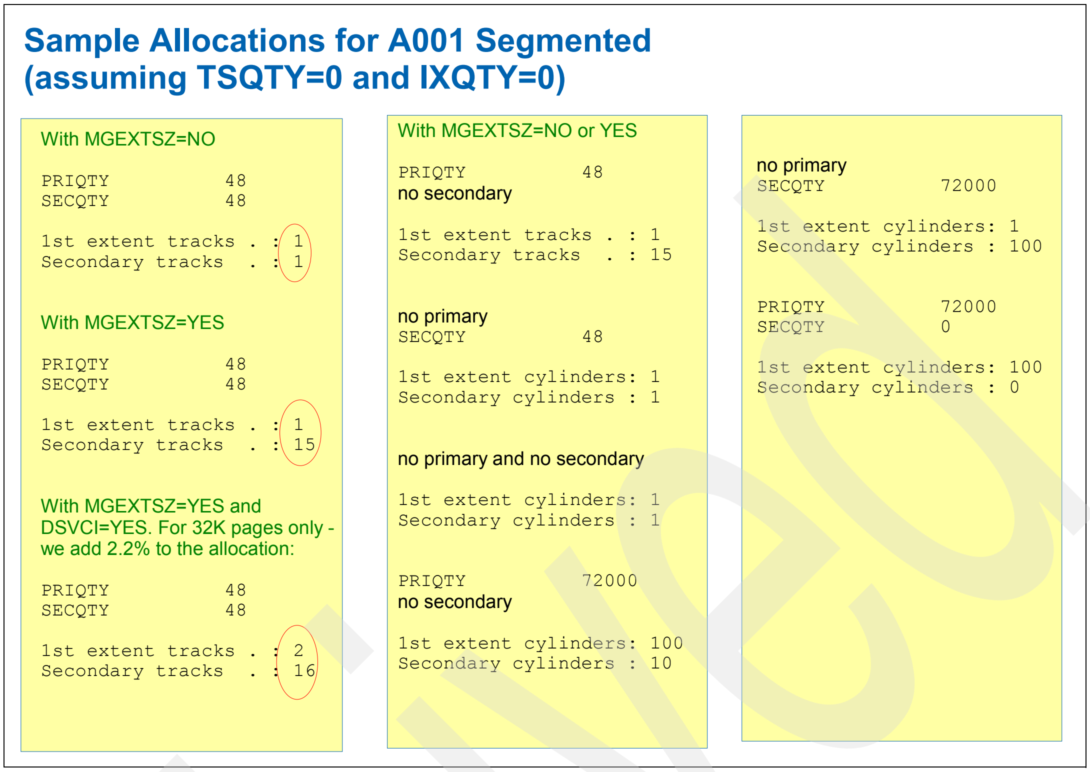
In the next two charts, we provide examples that summarize the data set allocations for a segmented table space.

- In the first case (Figure 19 on page 31), TSQTY and IXQTY DSNZPARMs are set, or have been left default, to zero.
  - When MGEXTSZ is set to NO (default) and you have defined a PRIQTY and SECQTY of 48 KB, the allocation is one track for both primary and secondary allocation.

    DB2 honors the request in your CREATE statement; system allocation and sliding allocation are not active.
  - When MGEXTSZ is set to YES, and you have defined a PRIQTY and SECQTY of 48 KB, the allocation is one track for primary allocation, and it becomes 15 tracks for secondary allocation.

    DB2 activates the default sliding secondary allocation.
  - When MGEXTSZ is set to YES, and you have defined a PRIQTY and SECQTY of 48 KB, and also set DSVCI to YES, in the case of 32 KB pages, DB2 increases the allocation to a primary of two tracks and a secondary of 16.

    This is to allow the 16 KB block to span the track.
  - For both MGEXTSZ YES or NO, with PRIQTY 48 KB and no SECQTY, the primary allocation is one track, the secondary allocation is 15 tracks.
  - For both MGEXTSZ YES or NO, with no PRIQTY and SECQTY 48 KB, both primary and secondary allocations are one cylinder.
  - For both MGEXTSZ YES or NO, with no PRIQTY and no SECQTY, both primary and secondary allocations are one cylinder.
  - For both MGEXTSZ YES or NO, with PRIQTY 72000 KB and no SECQTY, the primary allocation is 100 cylinders, the secondary allocation is 10 cylinders.
  - For both MGEXTSZ YES or NO, with no PRIQTY and SECQTY of 72000 KB, the primary allocation is one cylinder, the secondary allocation is 100 cylinders.
  - For both MGEXTSZ YES or NO, with PRIQTY 72000 and SECQTY of 0, the primary allocation is 100 cylinders, the secondary allocation is 0.



## Sample Allocations for A001 Segmented (assuming TSQTY=0 and IXQTY=0)

With MGEXTSZ=NO

```
PRIQTY          48
SECQTY          48

1st extent tracks . : 1
Secondary tracks  . : 1
```

With MGEXTSZ=YES

```
PRIQTY          48
SECQTY          48

1st extent tracks . : 1
Secondary tracks  . : 15
```

With MGEXTSZ=YES and DSVCI=YES. For 32K pages only - we add 2.2% to the allocation:

```
PRIQTY          48
SECQTY          48

1st extent tracks . : 2
Secondary tracks  . : 16
```

With MGEXTSZ=NO or YES

```
PRIQTY          48
```
no secondary

```
1st extent tracks . : 1
Secondary tracks  . : 15
```

no primary

```
SECQTY          48

1st extent cylinders: 1
Secondary cylinders : 1
```

no primary and no secondary

```
1st extent cylinders: 1
Secondary cylinders : 1
```

```
PRIQTY          72000
```
no secondary

```
1st extent cylinders: 100
Secondary cylinders : 10
```

no primary

```
SECQTY          72000

1st extent cylinders: 1
Secondary cylinders : 100

PRIQTY          72000
SECQTY          0

1st extent cylinders: 100
Secondary cylinders : 0
```

*Figure 19 Allocations with no system setting*

- In the second case (Figure 20 on page 32), TSQTY and IXQTY DSNZPARMs are set to 3600 KB, equivalent to five cylinders.
  - When MGEXTSZ is set to NO (default) and you have defined a PRIQTY and SECQTY of 48 KB, the allocation is one track for both primary and secondary allocation.

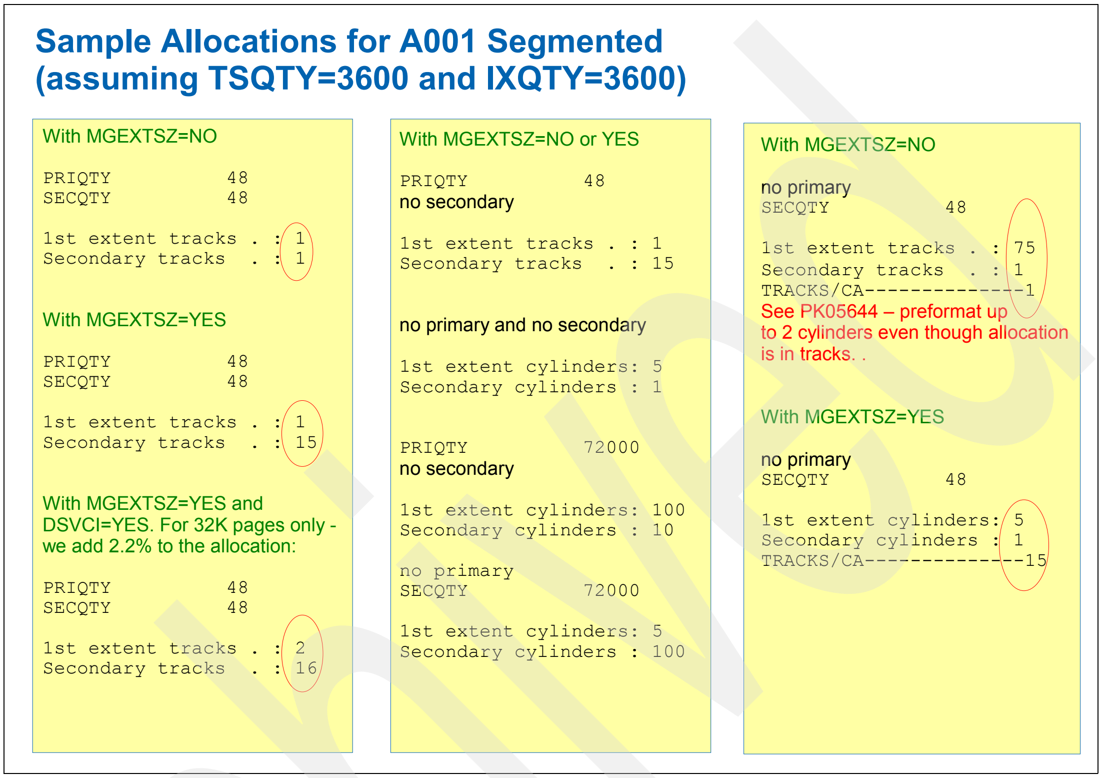
    DB2 honors the request in your CREATE statement; system allocation and sliding allocation are not active.
  - When MGEXTSZ is set to YES, and you have defined a PRIQTY and SECQTY of 48 KB, the allocation is one track for primary allocation, and it becomes 15 tracks for secondary allocation.

    DB2 activates the default sliding secondary allocation.
  - When MGEXTSZ is set to YES, and you have defined a PRIQTY and SECQTY of 48 KB, and also set DSVCI to YES, in the case of 32 KB pages DB2 increases the allocation to a primary of two tracks and a secondary of 16.

    This enables the 16 KB block to span the track.
  - For both MGEXTSZ YES or NO, with PRIQTY 48 KB and no SECQTY, the primary allocation is one track, the secondary allocation is 15 tracks.
  - For both MGEXTSZ YES or NO, with no PRIQTY and no SECQTY, the primary allocation is five cylinders, the secondary allocation is one cylinder.
  - For both MGEXTSZ YES or NO, with PRIQTY 72000 KB and no SECQTY, the primary allocation is 100 cylinders, the secondary is 10 cylinders.
  - For both MGEXTSZ YES or NO, with no PRIQTY and SECQTY 72000 KB, the primary allocation is five cylinders, the secondary is 100 cylinders.
  - For MGEXTSZ NO, with no primary and SECQTY 48 KB, the primary allocation is 75 tracks (five cylinders) and the secondary is one track. CA is now one track.

Notice that with APAR PK05644, instead of relying on the allocation unit, DB2 will preformat the space up to two cylinders when the page set is initialized or extended.

– For MGEXTSZ YES, with no primary and SECQTY 48 KB, the primary allocation is five cylinders and the secondary is one cylinder. CA is now 15 tracks.



Sample Allocations for A001 Segmented
(assuming TSQTY=3600 and IXQTY=3600)

With MGEXTSZ=NO

```
PRIQTY             48
SECQTY             48

1st extent tracks  . : 1
Secondary tracks   . : 1
```

With MGEXTSZ=YES

```
PRIQTY             48
SECQTY             48

1st extent tracks  . : 1
Secondary tracks   . : 15
```

With MGEXTSZ=YES and DSVCI=YES. For 32K pages only - we add 2.2% to the allocation:

```
PRIQTY             48
SECQTY             48

1st extent tracks  . : 2
Secondary tracks   . : 16
```

With MGEXTSZ=NO or YES

```
PRIQTY             48
```
no secondary

```
1st extent tracks  . : 1
Secondary tracks   . : 15
```

no primary and no secondary

```
1st extent cylinders: 5
Secondary cylinders : 1


PRIQTY             72000
```
no secondary

```
1st extent cylinders: 100
Secondary cylinders : 10

no primary
SECQTY             72000

1st extent cylinders: 5
Secondary cylinders : 100
```

With MGEXTSZ=NO

no primary
```
SECQTY             48

1st extent tracks  . : 75
Secondary tracks   . : 1
TRACKS/CA--------------1
```
See PK05644 – preformat up to 2 cylinders even though allocation is in tracks. .

With MGEXTSZ=YES

no primary
```
SECQTY             48

1st extent cylinders: 5
Secondary cylinders : 1
TRACKS/CA--------------15
```

*Figure 20   Allocations with system setting*

For a striped data set, the proposed extent size is divided by the number of stripes and each stripe can have up to 255 extents. Starting with z/OS 1.7, the 255 extent rule is lifted.

When extent consolidation is in effect in z/OS V1.5, the secondary space allocation quantity can be smaller than specified or calculated by the sliding scale based on the physical extent number. (See Table 4.) PTF UQ89517 for APAR PQ88665 corrects this problem by using the logical extent number together with the sliding scale.

*Table 4   Maximum allocation for sliding secondary extents - Without volume fragmentation*

| Max DS size in GB | Max alloc in cylinders | Extents to reach full size |
|---|---|---|
| 1 | 127 | 54 |
| 2 | 127 | 75 |
| 4 | 127 | 107 |
| 8 | 127 | 154 |
| 16 | 127 | 246 |
| 32 | 559 | 172 |
| 64 | 559 | 255 |

In theory, z/OS 1.7 allows VSAM LDSs to grow to 7,257 extents. Your Storage Administrator must set Extent Constraint Removal to YES in order to exceed the previous limit of 255 extents per component. Be careful of the number of extents, especially for 16 GB and 64 GB data sets, because they are near the maximum allowable extents for non-SMS managed LDSs, data sets allocated prior to z/OS 1.7, or data sets allocated with z/OS 1.7 where your Storage Administrator has set Extent Constraint Removal to NO.

Sometimes a Storage Administrator might allocate a special set of volumes in a Storage Group just for large data sets. Doing so typically enables large data sets to acquire larger chunks of free space while avoiding the volume fragmentation introduced by smaller data sets. Allocations to a volume in this special large Storage Group can be done through the Storage Group ACS routines using data set names of known large data sets or by a combination of size and data set pattern. For example, you can set up the Storage Group ACS routines to route only data sets above 400 MB of data and if the high-level qualifier is DB2PROD and the second qualifier is DSNDBD to a special large Storage Group. If your Storage Administrator is using this space method, beware that using the default PRIQTY may no longer work because for non-LOB data sets the allocation is one cylinder. In this case, you must be explicit about the data set name or add a PRIQTY.

## New CI sizes

With V7, DB2 uses only the VSAM CI size of 4 KB. If the page size is 8, 16, or 32 KB, DB2 treats the page as a chain of 2, 4, or 8 CIs. This requires VSAM to only allocate CIs in 4 KB blocks.

DB2 V8 introduces support for VSAM CI sizes of 8, 16, and 32 KB, activated by the new DSNZPARM parameter, DSVCI, option 6 on panel DSNTIP7. This is valid for both user-defined and DB2 STOGROUP-defined table spaces. With V8, index spaces use only 4 KB pages.

After you have activated the new CI sizes (in NFM), all new table spaces are allocated by DB2 with a CI corresponding to the page size. The table spaces already existing at the time of migration to DB2 V8 are later converted by the first execution of LOAD REPLACE or REORG. When using DB2 user-defined objects (USING VCAT), it is your responsibility to define the underlying VSAM cluster with the correct CISIZE.

DB2 uses VSAM LDSs for table space and indexes; they only use CI sizes that are a multiple of 4 KB, and no control interval definition fields (CIDFs). CIs are written by VSAM in physical blocks that try to use the best fit within the CI, and keep track occupancy decent.

Now, consider Table 5, which describes the existing relationship between DB2 V7 page sizes and VSAM CI sizes assuming a 3390 disk.

*Table 5 DB2 V7 - sizes in KB (usable track size 48 KB for 3390)*

| DB2 page size | VSAM CI size | VSAM physical block size | Blocks per track | DB2 pages per track |
|---|---|---|---|---|
| 4 | 4 | 4 | 12 | 12 |
| 8 | 4 | 4 | 12 | 6 |
| 16 | 4 | 4 | 12 | 3 |
| 32 | 4 | 4 | 12 | 1.5 |

VSAM does not know the DB2 page size, nor the fact that DB2 chains the VSAM CIs to make up its “logical” page size. VSAM I/Os cannot span cylinder boundaries or data set extent boundaries. So, we can see by Table 6 that if we have a DB2 page size of 32 KB, VSAM can write 22.5 32 KB DB2 pages per cylinder (15 X 1.5). A 32 KB DB2 page can therefore span a cylinder boundary and therefore span disk volumes. This is not a problem because DB2 chains the physical pages and guarantees the data integrity by ensuring all eight VSAM pages are written correctly.

A 32 KB DB2 “logical” page can span disk volumes, but restrictions exist in V7 that Concurrent Copy cannot support 32 KB pages, and striping of objects with a page size larger than 4 KB is not supported for DB2 data sets. In V7, striping is only allowed to 4 KB page sets and Concurrent Copy is not allowed for 32 KB table spaces. Concurrent Copy can copy the two volumes at different points in time; however, a single DB2 32 KB “logical” page can span these two disk volumes. A DB2 page should not be allowed to span a CA boundary and in the case of striping, a CI cannot span a cylinder.

Now, we see what has changed in V8. Consider Table 6, which describes the relationship between DB2 V8 page sizes and VSAM CI sizes.

*Table 6 DB2 V8 - sizes in KB (usable track size 48 KB for 3390)*

| DB2 page size | VSAM CI size | VSAM physical block size | Blocks per track | DB2 pages per track |
|---|---|---|---|---|
| 4 | 4 | 4 | 12 | 12 |
| 8 | 8 | 8 | 6 | 6 |
| 16 | 16 | 16 | 3 | 3 |
| 32 | 32 | 16 | 3 | 1.5 |

VSAM now deals with the four different CI sizes and, knowing the track size, now allows CI size equal to page size. VSAM writes the CI in a physical block of the same size as the page, but splits the 32 KB CI over two blocks of 16 KB in order not to waste space due track size (48 KB). It fits the 32 KB CIs 1.5 per track. VSAM also takes care of spanning track boundaries and operates I/O at the CI level. A 16 KB block is wasted every 15 tracks (CA size) because the CI cannot span a CA boundary. This is a small price to pay, now that VSAM is aware of the DB2 page size.

Now VSAM can guarantee that a DB2 page does not span CAs and therefore disk volumes. The new CI sizes reduce integrity exposures, and relieve restrictions on concurrent copy (of 32 KB objects) and the use of striping (of objects with a page size larger than 4 KB). It also resolves small integrity exposure with the Split Mirror solution using FlashCopy on 32 KB pages if at the extent border. A DB2 page now is always written in a single I/O and is totally contained in the same extent and the same device (even if striping is used).

## Allocation size for 32 KB pages with new CI size

Because a 16 KB block is not used every 15th track of a cylinder (one cylinder CA size) for 32 KB pages, you lose 2.2% of each cylinder. For DB2-managed data sets, DB2 will add 2.2% back into the allocation. For example, if you are creating a 32 KB table space with a PRIQTY of 72000 (which is the equivalent of 100 cylinders), with DSVCI=YES your allocation will be 103 cylinders. DB2 will recognize that you have requested a 32 KB page with the new CI size and automatically add the 2.2% overhead to your allocation that you are losing because of the new mechanism. Because a one-cylinder CA is used in this example, the 102.2 cylinders is rounded up to 103 cylinders. Example 2 on page 35 illustrates the added space with the results from LISTCAT.

*Example 2   LISTCAT output for 4 KB and 32 KB pages*

```
CREATE TABLESPACE
PRIQTY 72000

DB2 V7 32 KB LISTCAT output:
CISIZE--------------4096
PHYREC-SIZE---------4096
PHYRECS/TRK-----------12
SPACE-TYPE------CYLINDER
SPACE-PRI------------100

DB2 V8 4 KB LISTCAT output:
CISIZE--------------4096
PHYREC-SIZE---------4096
PHYRECS/TRK-----------12
SPACE-TYPE------CYLINDER
SPACE-PRI------------100

DB2 V8 32 KB LISTCAT output:
CISIZE-------------32768
PHYREC-SIZE--------16384
PHYRECS/TRK------------3
SPACE-TYPE------CYLINDER
SPACE-PRI------------103
```

## Performance with new CI sizes

What happens to I/O performance if we increase the VSAM CI size from 4 KB, to 8 KB, to 16 KB? Measurements show that large CIs are beneficial for FICON channels and storage controllers for table space scans. Table 7 shows I/O service times on an ESS 800 on a FICON channel for 4 KB, 8 KB, 16 KB, and 32 KB page sizes, respectively.

*Table 7   I/O service time on ESS 800*

| I/O service time on ESS 800 (ms) | Page sizes | | | |
|---|---|---|---|---|
| | **4 KB** | **8 KB** | **16 KB** | **32 KB** |
| **Cache hit** | 0.4 | 0.43 | 0.5 | 0.7 |
| **Cache miss** | 7.0 | 7.03 | 7.1 | 7.3 |

We see fewer “blocks” on a FICON link, which results in the improved performance. We can also see that changing the VSAM CI size has little impact on I/O performance.

However, there is an impact on DB2 buffer pool hit ratios. For example, if you increase your DB2 page size from 4 KB to 8 KB and do not increase your buffer pool size, then you are only able to cache half as many pages in the buffer pool. This can have an impact on the buffer pool hit ratio. If you do plan to use larger DB2 page sizes, we recommend that you also review your DB2 buffer pool sizes. Always ensure you have enough real storage available to back any increase in buffer pools, to avoid any paging to disk.

Now we turn our attention to DB2. Figure 21 on page 36 compares a DB2 table space scan with 4 KB, 8 KB, and 16 KB page sizes respectively.

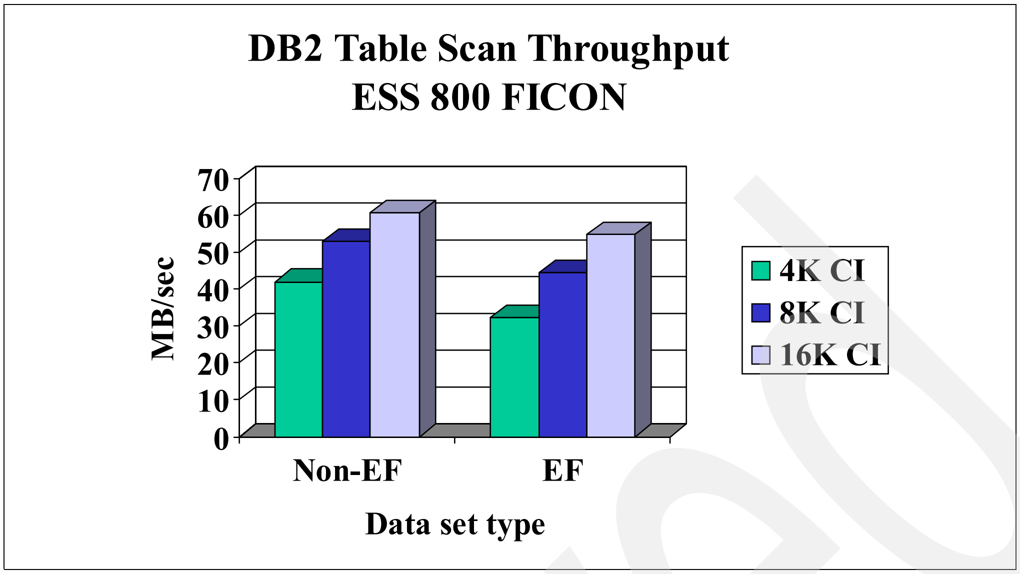


*Figure 21   Larger CI size impact on DB2*

A DB2 page size of 32 KB is not shown because VSAM implements this as two 16 KB CIs, so the results are the same as a DB2 16 KB page size. EF data sets are VSAM Extended Format (EF) data sets. EF data sets are required to allocate table spaces larger than 4 GB and for DFSMS striping.

The performance improvements are most pronounced when using EF VSAM data sets. We see 45% improvement when we increase the VSAM CI size from 4 KB to 16 KB, and we see huge 70% improvements if these data sets are also EF-Enabled. The EF to non-EF comparison shows better overall performance with non-EF data sets, but *this comparison changes dramatically with the new MIDAW protocol*. See “The MIDAW facility: improved channel architecture” on page 38.

### Conclusion for new CI sizes

All I/O-bound executions benefit with using larger VSAM CI sizes, including COPY and REORG. This enhancement can also potentially reduce elapsed time for table space scans.

There is little to be gained in performance by going above 16 KB CI size.

## Sequential processing and disk cache

SEQCACH is option 7 of the install panel DSNTIPE. It is used to determine whether data in the disk cache is to be used in sequential mode (BYPASS) or sequential detected mode (SEQ). Although BYPASS, the default, is the still a valid option for detected mode, cache has not been bypassed since the 3990 controllers were in use.

For the later disks (RVA, ESS, or DS8000), BYPASS uses sequential detect, and SEQ uses explicit command. The differences are:

- SEQ (explicit) puts tracks on the accelerated list and starts prestaging for next I/O. Operations are done on extent boundaries or stage groups. Explicit SEQ reacts faster and can end sooner than using the detect mechanism (BYPASS).
- BYPASS (sequential detect) stages data to the end of a stage group.

The recommendation is to set SEQCACH to SEQ.

## Utilities use of disk cache

SEQPRES is option 8 of the install panel DSNTIPE. It specifies whether (YES) or not (NO, default) *DB2 utilities* that scan a non-partitioning index followed by an update should allow data to remain in cache longer when reading data. If you specify YES, DB2 utility prefetch reads remain in cache longer, possibly improving performance of subsequent writes of large non-partitioned indexes.

Similar to what we discussed for sequential processing and disk cache (SEQCACH), the recommendation is to set SEQPRES to YES.

## Single volume archive

Starting with Version 6, DB2 has provided a *candidate* volume count on the dynamic allocation request for a new archive log data set allocated on a disk device. This was done to allow the data set to extend across multiple volumes, if needed, to reduce space-related allocation failures. DB2 is using the number of online storage volumes for the disk unit name defined in DSNZPARM (option 5 DEVICE TYPE 1 of install panel DSNTIPA), up to a maximum of 15 volumes.

Some users who manage the allocation of their archive log data sets outside of DB2 (for instance, with DFSMS) were having problems with the VOLUME count specified by DB2. Symptoms include allocation error MSGDSNJ103I with ERROR STATUS=970C0000, SMS REASON CODE=00004336 or REASON CODE=00004379 when guaranteed space was specified. Specifying guaranteed space can also result in wasted disk space as a primary extent is allocated on each candidate volume even though the data set does not actually extend to that volume. Additional problems were seen when there were not enough volumes in the SMS storage group to satisfy the request.

The solution to this problem came in the form of the DSNZPARM SVOLARC (option 16 of the install panel DSNTIPA). If you say YES, DB2 always specifies a UNIT and VOLUME count of 1 when allocating a new disk archive log data set. The default value of SVOLARC is NO, which allows the DB2 archive log data sets to extend across multiple volumes.

## Archive unit parameters

DSNZPARM values for DEVICE TYPE 1 and DEVICE TYPE 2 (specified with the two options on install panel DSNTIPA) allow for the DB2 archive log data sets and BSDS to be placed on different devices.

Many installations choose disk for both archive log copies. Some SMS-managed installations have a technique where they provide two separate Storage Groups, one for ARCHLOG1 data sets going to the first Storage Group and the other for ARCHLOG2 data sets going to the second Storage Group. DFSMShsm is then set up to archive the first Storage Group only once per day and the second Storage Group once an hour. The theory is that ARCHLOG1 and ARCHLOG2 data sets will not reside on the same migration level 2 (ML2) tape. The vast majority of installations mod (add) onto their ML2 tapes, which in today’s technology can house many gigabytes of data. The SMS separation profile does not apply to DFSMShsm tapes, and DFSMShsm does not have the capability to guarantee the separation of data sets. Because of this there have been many times when ARCHLOG1 and ARCHLOG2 data sets resided on the same tape. Availability is the issue at hand, and some concerns are:

- Tape can tear and become unusable.
- Tape can be misfiled by the Operator or Tape Librarian.
- Tape can be lost or destroyed.

The solution is to make sure that additional copies of the data sets are available. You can do that in several ways:

- Allow DFSMShsm to back up the archive data sets before migrating.
- Duplex your ML2 tapes. This may be an issue after the tapes are recycled, because they may once again reside on the same tape and become a single point of failure.
- Transmit a copy of at least one archive log and BSDS set to another site.
- Split UNIT and UNIT2 between two different device types. For example, UNIT goes to disk, and UNIT2 goes to tape.
- Use ABARS to copy one of the archives and BSDS from ML1 or ML2.

This can happen with any software dual copied data sets, including image copy data sets. This problem is not specific to DFSMShsm tapes; you might run into the same issues when using VTS (Virtual Tape Server) or DFSMS TMM (tape mount management).

## Utilities and FlashCopy

We have mentioned that DB2 integrates with z/OS 1.5 and later to deliver the system point-in-time recovery capability, a solution that is applicable to recovery, disaster recovery, or environment cloning needs. ESS and DS8000 FlashCopy is used for DB2 BACKUP and RESTORE SYSTEM utilities that implement the solution.

With DB2 V8 (APARs PQ92749 and PQ96956 added this function), when you run the CHECK INDEX utility with SHRLEVEL CHANGE, FlashCopy Version 2 (if available) will be invoked to make a fast copy of the data. CHECK INDEX will run against the shadow data set itself, thereby providing greater concurrency.

See *DB2 UDB for z/OS Version 8 Performance Topics*, SG24-6465, for more information about these functions.

Further integration between DB2 V9.1 for z/OS and z/OS 1.8 will allow tape support for BACKUP and RESTORE SYSTEM utilities, as well as the possibility of recovering single objects contained in the system backup.

## The MIDAW facility: improved channel architecture

The latest channel enhancement improves sequential access speed substantially for 4 KB page sizes on Extended Format or large table spaces. This improvement is available only on the System z9™ using the a new type of channel command. The MIDAW name stands for “modified IDAW.” An IDAW (indirect address word) is used to specify data addresses for I/O operations in a virtual environment.[1] The existing IDAW design enables the first IDAW in a list to point to any address within a page, but the chunk of storage must end on a page boundary. Subsequent IDAWs in the same list must point to the first byte in a page; also, all but the first and last IDAW in a list must deal with complete 2 KB or 4 KB units of data. This limits the usability of IDAWs to straightforward buffering of a sequential record.

CCWs with *data chaining* may be used to process I/O data blocks that have a more complex internal structure in which portions of the data block are directed into separate buffer areas. (This is sometimes known as scatter-read or scatter-write.) However, data-chaining CCWs

[1] There are exceptions to this statement and we skip several details in the following description. We assume that knowledgeable readers will merge this brief description with an existing understanding of I/O operations in a virtual memory environment.

are inefficient in modern I/O environments for reasons involving switch fabrics, control unit processing and exchanges, numbers of channel frames required, and so forth.

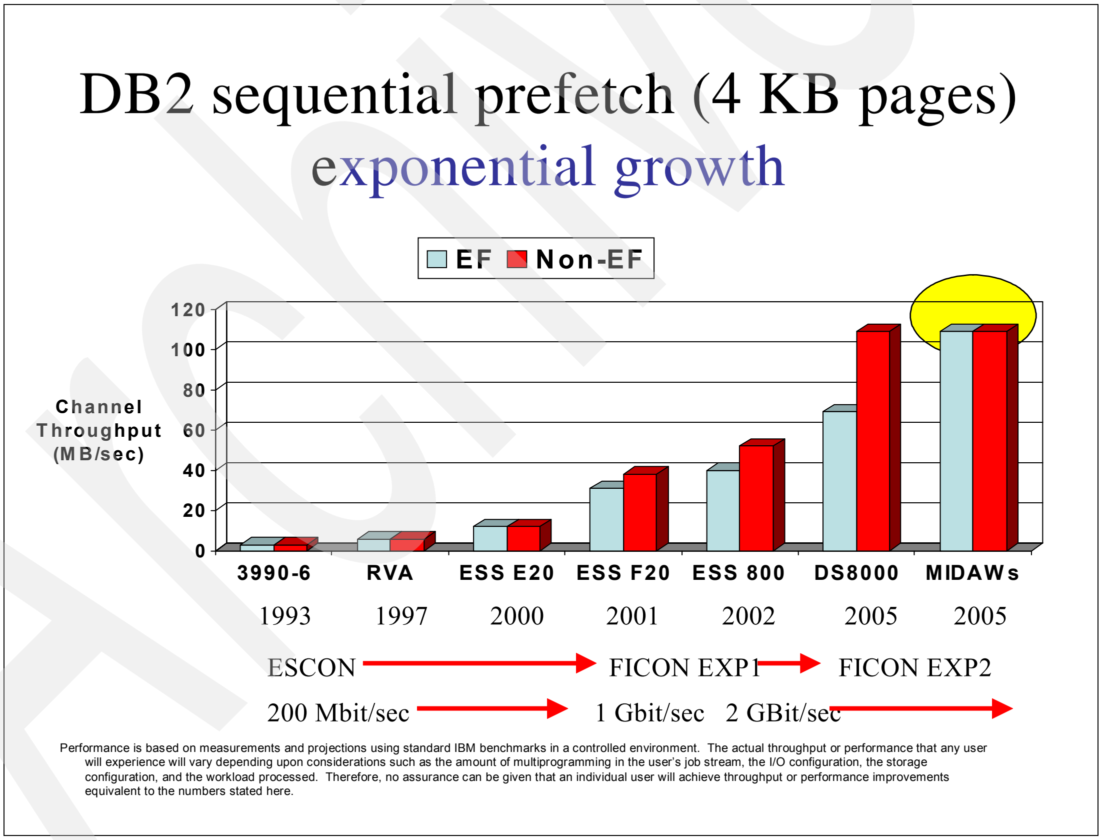
z/OS *extended format* data sets use internal structures (usually not visible to the application program) that require scatter-read (or scatter-write) operation.

IDAWs could not be used to do scatter reads or gather scatter writes. Data chaining with CCW was the old solution, but this produces less than optimal I/O performance, being too slow for FICON channels. MIDAWs enable scatter reads and gather writes with better performance.

The MIDAW function, which is new with the z9-109 server, provides the change of protocol for gathering data into and scattering data from discontinuous storage locations during an I/O operation. It improves FICON performance, especially when accessing DB2 databases, by reducing channel utilization and increasing throughput for parallel access streams.

Note that there is no change to ESCON, FICON, or control unit implementations. However, there is a greater chance that more *information units* (up to 16) may be sent to FICON control units in a single burst.

Several recent IBM Lab measurements concern MIDAW performance with DB2. Figure 22 illustrates a specific measurement that dealt with DB2 sequential prefetch of 4 KB pages.



Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the numbers stated here.

*Figure 22 MIDAW measurement for sequential prefetch of 4 KB pages*

From this set of measurements for sequential prefetch of 4 KB pages. we can see that MIDAW takes full advantage of the FICON Express2 capabilities, bringing the *channel throughput* up to more than 100 MB per sec. We also observe that MIDAW has eliminated the previous different behavior between *EF and non-EF* data set (Figure 21 on page 36) and it allows comparable throughput. DB2 requires the data sets to be defined in EF to enable striping or sizes larger than 4 GB.

Other sets of detailed measurements have been implemented by the Silicon Valley Lab and will be published in a separate technical document. From those measurements, we can draw the following conclusions:

- Block size effects for sequential reads

  MIDAWs minimizes the relative performance of different block sizes for sequential reads. We have previously seen that block size can affect FICON channel performance and that larger CIs perform better. Large page sizes (recommended for LOBs) have experienced major performance improvements from FICON Express2 and the DS8000. For example, for Extended Format (EF) data sets, the measured response time was 1.9 ms for 4 KB pages and 1.2 ms for 16 KB pages. Using MIDAWs, the response time for 4 KB pages drops to 1.2 ms, and the 16 KB pages drops to 1.1 ms. Similar performance is measured with non-EF data sets. MIDAWs target the 4 KB pages, so DB2 really Large Objects (LOBs) do not further benefit from MIDAWs; small LOBs, like those used by SAP solutions, might.

- More benefits from page fixing

  DB2 V8 introduced the ability to use long-term page fixing to avoid the dynamic page fix costs. We have seen that page fixing I/O-intensive buffer pools reduces CPU (see *DB2 UDB for z/OS Version 8 Performance Topics*, SG24-6465). With the I/O speed increasing, it becomes important to use long-term page fixing even for elapsed times. With MIDAWs, the I/O elapsed time reduction is further increased.

- What about writes?

  Writes come in three flavors: update writes (typical when executing SQL or logging), record-level format writes (typically used by DB2 utilities), and track-level format writes (used by DB2 to preformat page sets or logs).

  The type of chaining used for update write channel programs and for preformatting is the same as for reads. MIDAWs affect the performance of update writes and preformatting in the same manner that they affect reads.

- DB2 logging

  DB2 logs always use 4 KB pages. For extremely massive sequential inserts, logging might become an I/O bottleneck with large row sizes and infrequent commits. Small row sizes might stress first the log latch and the CPU. Measurements were done with 4000 byte rows sequential inserts generating large amount of log data with minimal CPU time.

  Prior to MIDAWs, the maximum log throughput using the DS8000 and FICON Express 2 was 84 MBps, and striping the log increased the bandwidth only slightly. MIDAWs increased the log bandwidth by 31%, reaching 116 MBps with two stripes.

- DB2 utilities

  DB2 utilities tend to be CPU bound, but at the same time they do a lot of sequential I/O. In this case MIDAWs help reduce the FICON channel utilizations, not necessarily the elapsed time. Two typically not-CPU-bound DB2 utilities are the Copy utility and the Recover utility. With MIDAWs, the throughput of both utilities with data sets of 4 KB page size shows an increase of about 10% for non-EF data sets and 40% or more for EF data sets.

- Striping for DB2 data sets

  In general, the value of using DFSMS to stripe data sets with 4 KB records has been somewhat limited by the performance problems of EF data sets. MIDAW removes these limits and increases the performance advantages of DFSMS striping.

  However, the value of striping for DB2 table spaces is still limited. This is due to the size of a DB2 V8 prefetch I/O, limited to 128 KB for queries. If there are two stripes, the 128 KB is split into two smaller I/Os of 64 KB. The measured elapsed time of a table scan using two stripes is 25% less than one stripe, not the expected 50% reduction. DB2 parallel table scans still outperform DFSMS striping, so whenever partitioning is possible, it is still the preferred solution for optimizing the performance of I/O-bound queries and utilities.

Sequential inserts are a special type of workload that benefits from striping. In order to optimize the performance of sequential inserts, it is beneficial to stripe both the DB2 log and the table spaces.

> **Note:** If your environment has a mix of MIDAW and non-MIDAW devices, PTFs UA25366 for HDZ11J0 and UA25367 for HDZ11K0 are recommended.

# Frequently asked questions about disks and SMS

This section provides answers to DB2-related frequently asked questions on disks and SMS.

## I hear that DB2 does not work with SMS; is that true?

When SMS was first introduced many years ago there were some recommendations not to use SMS for DB2 data sets. This recommendation was rescinded about a year or two after SMS was first announced. So, yes, you can fully use DB2 with SMS.

## I hear that only my DB2 user data can be SMS managed, not the system data; is that true?

The short answer is no. All DB2 data can be SMS managed. As a matter of fact, if you want to use DB2 V8 system level backup and restore, SMS is required for user and system data sets.

## If my volumes are SMS managed, do I still need to worry about space?

Yes. Your likelihood of getting that 2 a.m. call or visit to your executive’s office for a chat regarding an outage is greatly reduced. However, there are some issues to deal with:

- Is the space you were provided sufficient?
  - Are there any unexpected increases that now make it insufficient?
  - How do I track this now?
  - Who deals with these issues, the Storage Administrator or me?
- OK, I am happy with the space I have, but am I getting the disk performance that will make my customer happy?

These issues are highly dependent on your relationship with your Storage Administration group.

## My volumes are SMS managed; how can I tell what my volume names are, how much space I have left, and how much space I have used?

First, find out what your Storage Group names are. Then, to find out what the volume names are, you might try these suggestions:

- Use ISMF; be sure you are signed in as a Storage Administrator. Use option 0 for this task.
- Use option 6 - Storage Group, enter the Storage Group, and enter LISTVOL for a command.

Example 3 shows sample output.

*Example 3 ISMF Storage Group LISTVOL report*

```
VOLUME FREE       %     ALLOC      FRAG   LARGEST    FREE
SERIAL SPACE      FREE  SPACE      INDEX  EXTENT     EXTENTS
-(2)-- ---(3)---  (4)-  ---(5)---  -(6)-  ---(7)---  --(8)--
CBSM01   2598244    94     173256     71    2112449       17
CBSM02   2408773    87     362727     40    2223675       24
CBSM03   2566481    93     205019     39    2327430       16
```

If you know the volser, you can use ISPF 3.4 with option V. You will not see Storage Group information, just volume information. See Example 4.

*Example 4 ISPF 3.4 volume space listing*

```
Volume . : CBSM02

Unit . . : 3390

 Volume Data             VTOC Data               Free Space   Tracks    Cyls
 Tracks . :   50,085     Tracks  . :        12   Size  . . :  43,530   2,898
 %Used  . :       13     %Used . . :        12   Largest . :  40,185   2,679
 Trks/Cyls:       15     Free DSCBS:       531   Free
                                                 Extents . :      24
```

You can also use DCOLLECT through ISMF or execute IDCAMS. A sample REXX for the output is available in ISMF under **Enhanced ACS Management** → **SMS Report Generation**.

If you have SDSF authority to issue MVS commands, you can display the Storage Group volumes as shown in Example 5.

*Example 5 MVS command to display Storage Group volumes*

```
D SMS,SG(CB390),LISTVOL
IGD002I 10:54:27 DISPLAY SMS 404

STORGRP  TYPE    SYSTEM= 1 2 3 4 5 6 7 8
CB390    POOL            + + + + + + + +

VOLUME   UNIT    SYSTEM= 1 2 3 4 5 6 7 8                 STORGRP NAME
CBSM01   9025            + + + + + + + +                   CB390
CBSM02   9034            + + + + + + + +                   CB390
CBSM03   9035            + + + + + + + +                   CB390
***************************** LEGEND *****************************
. THE STORAGE GROUP OR VOLUME IS NOT DEFINED TO THE SYSTEM
+ THE STORAGE GROUP OR VOLUME IS ENABLED
- THE STORAGE GROUP OR VOLUME IS DISABLED
* THE STORAGE GROUP OR VOLUME IS QUIESCED
D THE STORAGE GROUP OR VOLUME IS DISABLED FOR NEW ALLOCATIONS ONLY
Q THE STORAGE GROUP OR VOLUME IS QUIESCED FOR NEW ALLOCATIONS ONLY
> THE VOLSER IN UCB IS DIFFERENT FROM THE VOLSER IN CONFIGURATION
SYSTEM  1 = SYSA       SYSTEM  2 = SYSB       SYSTEM  3 = SYSC
SYSTEM  4 = SYSD       SYSTEM  5 = SYSE       SYSTEM  6 = SYSF
SYSTEM  7 = SYSG       SYSTEM  8 = SYSPLEX1
```

## How many Storage Groups should I have in my production environment?

Production should have its own storage group for DB2 table spaces and indexes separate from all other environments:

- Start with at least three separate Storage Groups. If you will use the DB2 V8 BACKUP and RESTORE SYSTEM (requires z/OS 1.5) you are required to have separate Storage Groups for the BSDS and active log data sets from all other data. The three categories are:
  - DB2 Catalog and Directory objects
  - BSDS and active log data sets
  - DB2 user data
- Discuss ICF catalog placement strategies with your Storage Administrator regarding different recovery scenarios.
- Place the sort (DSNDB07) data sets on a separate volume from the above if not using PAV and MA.

Newer disk devices that use PAV, MA, and so forth typically do not require data and indexes on separate volumes. However, older versions might benefit by the separation (still watch for hot spots). For separate Storage Groups:

- Use a unique data set naming convention separating data and indexes that will be resolved by the Storage Group ACS routine for correct placement.
- Use ZPARM values for SMSDCFL and SMSDCIX, in macro DSN6SPRM. These hidden DSNZPARM values provide SMS with one Data Class for data (SMSDCFL) and another for indexes (SMSDCIX) that will be resolved by the Storage Group ACS routine for correct placement.

Provide a Storage Group for your archive log data sets.

- Recovery using archive log from disk is much faster than tape for parallel recoveries where several logs reside on the same tape. The exception to this is when archive logs are stored on DFSMShsm tapes (aside from recall time). In this case, make sure the Storage Group can handle additional logs.

Provide a Storage Group for image copy data sets. Recoveries from image copy data sets residing on disk will be much faster for parallel operations because there is no need to serialize image copy data sets if stacked on the same tape.

Determine realistically how many archive log and image copy data sets are required for recovery situations, and size the volumes in your Storage Groups accordingly.

## How many Storage Groups should I have in non-production?

For non-production, it depends on the requirements for:

- Performance
- Backup and recovery
- Types of environments: sandbox, development, test, ERP and non-ERP, and so on
- Amount of data in each environment

Separation of environments depends on business requirements. You might want to stay with the production strategy, or you might want to combine some or all of the Storage Groups for the above environments.

## Consolidating to large volumes - Mod 27 or 54

My Storage Administrator just told me that they are putting in mod 54s, and instead of my current 15 mod 3 volumes, they are trading me up to one volume with the capacity of a little bit more than 19 mod 3s. That sounds like a great deal, is there anything I need to consider?

- Although VSAM data sets can have up to 255 extents (increased in z/OS 1.7), there is a limitation of 123 extents per volume. This means that extending past 123 extents will not be possible with just one volume, so you will never grow beyond this point. This is true for other object types as well, such as extended format sequential data sets, which similar to VSAM will allow 123 extents per volume.
- Are you currently backing up (full volume dump) your volumes? There will be a lack of parallelism. There will be one very large dump instead of up to 20 dumps in parallel. How long will this function take? This is true for other operations, such as DEFRAG.
- Consider the amount of time it takes to FlashCopy a volume. Again, similar to the dump issue, your copies will not be in parallel.

## How did these space allocations happen?

I created an image copy data set as SPACE=(CYL,(1000,100)). I expected 1400 cylinders and five extents when multi-volume, and I got 2300 cylinders with five extents. What happened?

- If your data set uses the guaranteed space attribute, then SMS is working as designed. In this scenario the primary allocation is propagated to the next volume when it went multi-volume. Extents are as follows - VOL1=1000,100 VOL2=1000,100,100.

I image-copied an 800-track table space by mistake. The output was TRK(1,1), but the allocation worked, how did that happen?

- If your Storage Administrator assign EF for your image copy data sets and if there are at least seven volumes with enough free space, the data set can spread up to 123 extents (as opposed to 16 extents non-EF) on each volume, the end result will be a sequential data set with 800 extents. This is not a great way of doing business, but there is no outage. With z/OS 1.7 you can have roughly 7257 extents max for 60 volumes. Extent consolidation with SMS is also a possibility.

## My Storage Administrator is seeing a lot of disk write bursts. Is there anything I can do to help?

Your Storage Administrator will see this in an RMF Cache Volume Detail report as “DFW Bypass.” For newer disks, this is actually a DASD Fast Write retry and no longer a bypass.

This means that NVS (Non Volatile Storage) is being flooded with too many writes. The controller will retry the write in anticipation that some data in NVS has been off-loaded.

For RAMAC devices, the solution was to lower VDWQT to 0 or 1.

- This will cause high DBM1 SRM CPU time.
- May no longer be needed for ESS and DS8000 devices. Test and verify settings.

For very large buffer pools with many data sets, consider lowering VDWQT to 0 or 1.

- Also might work well for ESS and DS8000. The trade-off is still higher DBM1 SRM CPU time.

- Test and retest! Validate such things as Class 3 times.

## Do I need PDSEs?

I have heard about PDSEs. What are they, and do I need them? Do they have to be SMS managed?

PDSEs are SMS-managed data sets (non-SMS possible, partial APAR list - OW39951).

Before DB2 V8, there were no requirements for PDSEs. The following are delivered as PDSEs starting with V8:

- ADSNLOAD
- ADSNLOD2
- SDSNLOAD
- SDSNLOD2

Recommendation: Use PDSEs for load libraries that contain stored procedures. This reduces the risk of out-of-space conditions due to adding or updating members. This is true for DB2 V7 as well.

PDSEs are like PDSs, but much better:

- Up to 123 extents (instead of 16 for PDSs). Cannot extend beyond one volume.
- Number of directory blocks is unlimited.
- Does not require periodic compression to consolidate fragmented space for reuse.
- There is no need to recreate PDSEs when the number of members expands beyond the PDS's available directory blocks.

## Space anomalies: Question 1

How did this happen? I have 90 cylinders primary, 12 cylinders secondary, 63 extents, but 25530 tracks allocated. See Example 6.

*Example 6   Report of space anomalies - Question 1*

```
ISPF 3.4 listing                    Tracks  XT  Device
---------------------------------------------------------
RI1.DSNDBD.A020X7KR.CKMLPR.I0001.A001  25530  63  3390

LISTCAT output:
ALLOCATION
        SPACE-TYPE------CYLINDER     HI-A-RBA------1254850560
        SPACE-PRI-------------90     HI-U-RBA------1113292800
        SPACE-SEC-------------12

primary+(secondary*(extents-1))=space
90+(12*(63-1))=834 cylinders, 12510 tracks, not 25530!
```

Here is how the scenario unfurled:

1. CREATE TABLESPACE PRIQTY 64800 SECQTY 8640.
2. After some time and space, ALTER TABLESPACE SECQTY 20000.
3. Add more rows, trip extents.
4. No REORG afterward.

5. DB2 information correct, but MVS information is not!
6. CYL(90,12) with 63 extents, which should be 2510 tracks. The exception to this is when you have additional extents due to volume fragmentation.

Let your Storage Administrator know that what they see is not always what they get!

This case is not a disk fragmentation issue. After the ALTER, DB2 knows the allocation converted to CYL(90,28). However, MVS still thinks it is CYL(90,12) until redefined by a process such as REORG without a REUSE. PRIQTY and SECQTY is actually what DB2 uses, not CYL(90,12).

## Space anomalies: Question 2

I understood the last chart, but how did I actually get *less* space than I requested?

- When your Storage Administrator has set up a Data Class with the space constraint relief attribute on, and with the request for a percentage for space reduction, your data set allocated can actually be less than requested. This can happen if your volume does not have enough space.
- For example, a 4 KB object, created with PRIQTY 72000 (100 cylinders), the Data Class space constraint was set up to allow 10% reduction, you had one volume with 92 cylinders remaining.
- Results:
  - The DB2 catalog will still show the equivalent of PRIQTY 72000.
  - The actual MVS allocation will be 90 cylinders or the equivalent of PRIQTY 64800.

## What about catalogs?

When Storage Administrators talk about catalogs, are they talking about the DB2 catalog?

Generally, the answer here is no. Storage Administrators view the term catalog as the ICF catalog, which they typically maintain. Make sure that when you or your Storage Administrator use the term catalog it is specifically stated which one. This will avoid needless confusion, errors, and arguments.

## Do I need to let my Storage Administrator know anything about DB2 V8?

I am migrating my DB2 to V8, is there anything I need to tell my Storage Administrator?

It depends on how your Storage Administrator set up the ACS routines for the DB2 data set names. If they are looking at the low-level qualifier of your DB2 data set name and you plan on using partitions above the V7 limit of 254, then the answer is yes. The LLQ in DB2 V8 will have the following pattern:

- A001-A999 for partitions 1 through 999
- B000-B999 for partitions 1000 through 1999
- C000-C999 for partitions 2000 through 2999
- D000-D999 for partitions 3000 through 3999
- E000-E096 for partitions 4000 through 4096

Your Storage Administrator may need to change some reporting programs as well.

## What is extent reduction?

My Storage Administrator told me they re-created my DB2 data sets with high extents. They are now as low as one extent per data set. Are there any issues?

Storage Administrators have a number of ways of causing extent reduction that potentially bring back a data set in one extent, among them:

- DFSMShsm MIGRATE and RECALL functions
- DFSMSdss COPY or DUMP and RESTORE functions
- DEFRAG with the CONSOLIDATE keyword

Using such functions as DFSMSdss COPY might be much faster than running REORGs.

Do you have SQL that uses the DB2 catalog to report on extents? This can potentially be a problem. You might be redoing REORGs unnecessarily because the move was done outside of DB2 and DB2 does not know about it.

> **Note:** The EXTENTS column in the RTS will be updated only when an update or applicable utility is run for the object. A simple start after extent reduction or a read based on SELECT will not update the EXTENTS column (same issue as the catalog).

Do you use high extents as a tool to review issues with clustering indexes? This can potentially be a problem. Review CLUSTERRATIOF more closely.

## How much data can I really lose?

Is it possible to lose just one or a few volumes with newer disk?

Yes, although it is extremely rare to lose just one or a few volumes instead of an entire disk controller's worth.

Recommendation: Because of this (there are other reasons), I run a daily disk report by volume for my DB2 objects. Some things to think about if something does go wrong:

- What was on the volume I lost?
  - All indexes? Maybe I can just rebuild them.
  - If part of one application or part of a partition data set, it might not be too bad.
  - My new mod 54 with *all* of my data? Find out what your alternatives are. Hopefully, there are some based on the architecture you have built in for this type of event.
  - Otherwise, this gets into a much bigger discussion that we do not have time for now.
- So, do I care if my data sets are multi-volume?
  - Yes, based on the information above. Lots of multi-volume data sets can cause you lots of additional restores if a piece of your data resided on the crashed volume.

## What about pseudo deletes?

My index keeps on growing and tripping extents, even after deletes. What is wrong with DB2? How can I control the extent growth?

This one is actually a DB2 issue concerning pseudo deleted entries in your index, not a Storage Management issue:

- For an index, deleted keys are marked as pseudo deleted.

- Actual cleaning up will not occur except during certain processes (for example, before a page split).
- High CPU cost of an index scan: Every time an SQL statement makes a scan of an index, it has to scan all entries in the index. This includes pseudo deleted entries that have not yet been removed.
- You can calculate the percentage of RIDs that are pseudo deleted based on the values of PSEUDO_DEL_ENTRIES and CARDF in SYSINDEXPART:
  - (PSEUDO_DEL_ENTRIES/CARDF)*100

Recommendation: REORG INDEX, if the percentage of pseudo deleted entries is greater than 10% for non Data Sharing and 5% for Data Sharing.

## What are FlashCopy and SnapShot?

I hear about SnapShot and FlashCopy versions 1 and 2. What is the difference between all of these (at a very high level)?

- SnapShot (RVA only)
  - SnapShot can quickly move data from the source device to the target device.
  - Data is “snapped” (quickly copied) directly from the source location to the target location.
- FlashCopy (ESS and DS8000) - versions 1 and 2
  - FlashCopy V1 requires the entire source volume and target volume to be involved in a FlashCopy relationship. FlashCopy V1 relationships do not allow any other FlashCopy relationships to exist on either the source or target volume.
  - FlashCopy Version 2 enhances the FlashCopy function by providing an alternative method to copying an entire source volume to a target volume:
    - Multiple FlashCopy relationships are allowed on a volume.
    - Track relocation is possible because when copying tracks, the target tracks do not have to be in the same location on the target volume as on the source volume.
    - The restriction that a FlashCopy target and source volume must be in the same logical subsystem (LSS) in an ESS is removed. However, FlashCopy must still be processed in the same ESS.
    - V2 is 10 times faster than FlashCopy V1.

# The team that wrote this Redpaper

This Redpaper was produced by a team of specialists from around the world working at the International Technical Support Organization, San Jose Center.

**Paolo Bruni** is a DB2 Information Management Project Leader at the ITSO, San Jose Center. He has authored several Redbooks™ about DB2 for z/OS and related tools, and has conducted workshops and seminars worldwide. During Paolo’s many years with IBM, previously as employee and now as a contractor, his work has been mostly related to database systems.

**John Iczkovits** is a Consulting IT Specialist with IBM Advanced Technical Support (Dallas Systems Center) Americas. He provides DB2 for z/OS technical support and consulting services. His areas of expertise includes DB2 data sharing, performance, and availability. His work for IBM includes experience supporting DB2 as a Systems Engineer, Database

Administrator, IT Specialist, consultant, and project lead. He has an operations background and 21 years of IT experience, ranging from database products such as DB2 and IMS™ to MVS systems support, including storage management. He has co-authored IBM Redbooks, white papers, and presented at SHARE, the DB2 Tech Conference, and local DB2 user groups.

Thanks to the following people for their contributions to this project:

Rich Conway
Emma Jacobs
Bob Haimowitz
Sangam Racherla
Betsy Thaggard
International Technical Support Organization

Jeff Berger
John Campbell
Julie Chen
Roger Miller
Akira Shibamiya
Jim Teng
IBM Silicon Valley Laboratory

Rick Butler
BMO Financial Group, Toronto, Canada

Bart Steegmans
IBM Belgium

Charlie Burger
Advanced Technical Support, Americas

Joe Hyde
Lee La Frese
IBM Tucson Laboratory

# Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
*IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.*

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Send us your comments in one of the following ways:

- Use the online **Contact us** review redbook form found at:
  **ibm.com**/redbooks
- Send your comments in an e-mail to:
  redbook@us.ibm.com
- Mail your comments to:
  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400 U.S.A.

## Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

DB2®
DFSMSdfp™
DFSMSdss™
DFSMShsm™
DFSMSrmm™
DS8000™
Enterprise Storage Server®
ESCON®
FlashCopy®
FICON®
IBM®
IMS™
MVS™
RAMAC®
Redbooks™
Redbooks (logo) ™
RMF™
System z9™
TotalStorage®
z/OS®
z9™
zSeries®

Other company, product, or service names may be trademarks or service marks of others.