DB2 for z/OS V9.1 Bufferpool Handling

Begriffe

Address Space	Basierend auf 64-Bit-Adressierung hat nun jeder Address Space einen theoretischen Adressraum von bis zu 16 EB (Exabyte), "the beam" genannt. Trotzdem gelten Einschränkungen: • Maximum von 16 MB "below the 16 MB line" • Maximum von 2.032 MB "above the 16 MB line" and "below the 2 GB bar" • Hinzu kommt nun der Adressraum über 2 GB „above the bar"
Virtual Storage	DB2 benötigt unterhalb der 16-MB Line weniger als 40 KB für jedes Subsystem und weniger als 24 KB für jedes IRLM Subsystem. Der größte Teil an DB2 Common Data liegt innerhalb der Extended Common Service Area (ECSA). Die meisten Modules, Control Blocks und Puffer liegen in der Extended Private Area oberhalb der 16-MB- oder der 2-GB Grenze. DB2 V9.1 for z/OS benötigt 128 GB an 64-bit Shared Storage für jedes Subsystem. Der DB2 V9.1 Address Space kann sich theoretisch auf 1 Exabyte ausdehnen.
Getpage	Eine DB2 GETPAGE Request Operation entsteht immer dann, wenn Table- oder Index-Daten aus einer Page für die Durchführung eines SQL Statements angefordert werden. DB2 überprüft dabei ob sich die Page bereits im Bufferpool befindet. Falls nicht, wird mit einer I/O Operation die Page von Disk gelesen, oder in einem Data Sharing Verband, vom Group Bufferpool angefordert. Der Aufwand für Getpage Request kann in DB2 Monitoren klassifiziert werden: • Wenn die „other read I/O time" nahe bei der „total query elapsed time" liegt, dann ist das SQL Statement I/O-intensive. „Other read I/O time" ist die Zeit, die DB2 auf Pages wartet, die in den Bufferpool gelesen werden • Wenn die „CPU time" nahe bei der „total query elapsed time" liegt, dann ist die Query Processor-intensive. • Wenn die Processor Time zwischen 30% and 70% der Elapsed Time liegt, dann kann ein SQL Statement als ausgewogen zwischen CPU und I/O betrachtet werden. Im weiteren Verlauf des Dokuments werden die verschiedenen Getpage Request Typen erörtert: • Random • Sequential • List
Virtual Bufferpool	Bufferpools werden unter dem DSNDBM1 Primary Address Space "above the bar" (2 GB) allokiert und können in der Summe je Subsystem bis zu 1 TB (seit DB2 V8) umfassen. Die Speicherplatzbedürfnisse werden durch den Hauptspeicher und Auxiliary Storage abgedeckt. Die Allokierung des Speicherplatzes findet mit der erstmaligen Nutzung eines Objekts, das dem Bufferpool zuordnet ist, statt. Eine spezielle Art von Bufferpool wird nur im Parallel Sysplex Data Sharing benutzt. Es handelt sich um den Group Bufferpool, der innerhalb des Coupling Facilities angesiedelt ist und verschiedene DB2 Subsysteme in die Lage versetzt, Informationen zu teilen und dabei die Konsistenz gewährleistet. DB2 begrenzt die Größe eines Bufferpools auf ungefähr das doppelte an verfügbarem Real Storage. Um Paging zu vermeiden sollte der gesamte Bufferpool die Größe des DB2 zur Verfügung stehenden Real Storage nicht überschreiten. Bufferpool Minima:

	4 KB = 2.000 Pages 8 KB = 1.000 Pages
	16 KB = 500 Pages 16 KB = 250 Pages
	
	Buffer Pool Enhancements in V8
	• 64-bit Buffer Pools
	• Batch Group Buffer Pool Writes
	• Miscellaneous Enhancements
	
	Buffer Pool Enhancements in V9
	• Automatic Buffer Pool Management
	• Workfile Buffer Pools
	• Commands to Open/Close Tablespace and Index
	• Miscellaneous Enhancements
	• Größere Prefetch und Deferred Write Quantities:
	  • Max von 128 KB in V8 ➔ 256 KB in V9 in einem SQL Table Scan
	  • 256 KB in V8 ➔ 512 KB in V9 in Utilities
	• Größere Pools für ...
	  • Sequential Prefetch, if VPSEQT*VPSIZE> 160 MB für SQL, 320 MB für Utilities
	  • Deferred Write, if VPSIZE> 160 MB für SQL, 320 MB für Utilities

Hiperpool und Dataspace	Diese obsoleten Einrichtungen stellten bis V8 optionale Erweiterungen der Bufferpools dar und konnten die ursprüngliche maximale Ausdehnung des DB2 Address Space erwirken. Hiperpools befanden sich lediglich im Expanded Storage, waren Cache der Bufferpools und konnten keine modifizierten Pages enthalten.
Automatic BP Manage-ment	Automatic Bufferpool Management (seit DB2 V9.1) ersetzt nicht das herkömmliche Bufferpool Tuning, sondern ergänzt es insbesondere im Hinblick auf Größen-änderung des Pools. DB2 ändert dabei, abhängig von Informationen des Real-Time Workload Monitoring, die Größe um bis zu 25% der Ausgangsgröße. Mit dem Befehl ALTER BUFFERPOOL AUTOSIZE(YES) wird das Feature aktiviert.
Buffers	Es wird nach folgenden Buffers unterschieden: • Free Buffers • Active Buffers • ‚being read': durch Anwendung angefordert • locked: non-stealable • not locked: stealable • ‚write pending': page durch DML verändert (‚*in-use*') • non-stealable, solange nicht auf DASD geschrieben • Variation: Locked+'not being read'+'not write pending', heißt, daß eine veränderte Page bereits auf DASD geschrieben wurde. stealable Die weitere Kategorisierung erfolgt nach der Nutzung der Buffer:

	• Random Buffers: Index Probe, Index Look-Aside, OLTP, Small Tables, Joins of Smaller Tables or Smaller Result Tables • Sequential Buffers: Sequential Prefetch, Sequential Detection, DSNDB07 (Order, Group, Union etct, Merge Scan Joins)
Sync. I/O	Wird vorgenommen für Random Requests, oder Sequential Requests von geringen Datenmengen oder Sequential Prefetch Requests, die nicht befriedigt werden können.
Async I/O	Für die Prefetch Operationen (Sequential, Dynamic, List Sequential) werden s.g. Read Engines beauftragt, Daten asynchron zu lesen. Jede Read Engine kann, je nach Bufferpool-Größe bzw. zu 64 (bei Utilities 128) Pages auf einmal lesen.
Sync Write	Mit dieser Methode werden veränderte Bufferpool Pages auf DASD geschrieben, wenn der *Immediate Write* Threshold erreicht wurde. Der Write erfolgt synchron, was bedeutet, daß abhängige Threads warten müssen. Mit einer einzigen Write I/O können max. 32 (bei Utilities 64) Pages über eine maximale Spannweite von 180 Pages (1 Cyl. 3390 DASD) geschrieben werden.
Immediate Write	Ein Immediate Write kann aus folgenden Gründen veranlaßt werden: • 97,5% der Pages eines Bufferpools sind *‚in-use‘* (IMTH) • Close Dataset • Checkpoint • keine Write Engines verfügbar Der Immediate Write veranlaßt den *Sync Write* von BP Pages.
In-Use Pages	Sind veränderte, non-stealable Pages, für die ein Writing-Pending gilt. Diese Pages werden mit einem sync write oder async write auf DASD geschrieb, womit die Pages wieder frei/stealable werden.
Async Write	Im Gegensatz zum Sync Write können mit s.g. ‚Write Engines‘ asynchron zu laufenden Threads veränderte Pages auf DASD geschrieben werden. Die Write Engines können mit einer einzigen Write I/O max. 32 (bei Utilities 64) Pages über eine maximale Spannweite von 180 Pages (1 Cyl. 3390 DASD) geschrieben werden. Async Write werden veranlaßt, wenn die variablen Threshold DWQT oder VDWQT (Belegung durch veränderte Pages eines Pagesets) erreicht wurden
Bufferpool Page Stealing	Der Page Stealing Algorithm (PGSTEAL) ist mit DB2 V6 eingeführt worden und beschreibt mit welcher Methodik Pages aus dem Bufferpool entfernt werden. • LRU – last recently used (Default): Diese Option hält Pages im Bufferpool, die häufig benötigt werden und entfernt stattdessen unbenutzte Pages. Dieser Algorithmus stellt sicher, dass am meisten genutzten Pages immer im Bufferpool sind. • FIF (first in, first out): Diese Option entfernt die ältesten Pages, ungeachtet wie regelmäßig diese benötigt werden. Diese einfache Vorgehensweise resultiert in etwas geringerem Aufwand für eine Getpage-Operation und kann die DB2-interne Latch Contention in solchen Umgebungen reduzieren, die ein Höchstmaß an Konkurrenzverarbeitung verlangen.
Bufferpool long-term Page Fixing	Mit dieser Option (seit DB2 V8) fixiert DB2 Pool Pages im Real Storage, was insbesondere günstig für Objekte mit hoher I/O-Rate ist. Für Objekte mit wenig /I/O (z.B. für Indexes mit 100% Bufferpool Hit Ratio, oder Read-only Daten) wird diese Methode nicht empfohlen. Um zu vermeiden das PGFIX(YES) Bufferpools die Real Storage Kapazität sprengen, nutzt DB2 einen 80%-Grenzwert und stellt, wenn er überschritten würde, auf PGFIX(NO) um.

	Methode um sicherzustellen, dass alle Pages durch Real Storage abgedeckt werden: 1) ALT BPOOL(BPxx) VPSIZE(0) gefolgt von ... 2) ALT BPOOL(BPxx) VPSIZE(yyyy) PGFIX(YES)

Dynamische Grenzwerte - Thresholds

Abbrev.	Thres-hold Art	Beschreibung
IWTH	**fix** **97,5 %**	Immediate Write Threshold (97,5%) Wenn dieser Grenzwert von 97,5% nicht verfügbarer Pages des BP errreicht ist, werden alle veränderten Pages sofort synchron aus DASD geschrieben. Diese Situation kann für BP Pages ebenfalls durch einen QUIESCE WRITE(YES) auftreten, oder wenn veränderte Pages nach mehr als 2 System-Checkpoint noch nicht auf DASD geschrieben wurden.
DMTH	**fix** **95%**	Data Management Threshold (95%) Sofern dieser Grenzwert überschritten wird, verändert DB2 seine Art und Weise BP Pages zu verarbeiten: Je angeforderter Row wird eine Page gelesen bzw verändert, was die Performance satzweiser Verarbeitung größerer Datenmengen erheblich verschlechtern kann.
SPTH	**fix** **90%**	Sequential Prefetch Treshold (90%) Fester Grenzwert, der besagt, daß bei einer Nutzung von über 90% aller BP Pages der Sequential Prefetch ausgesetzt wird, bis erneut Speicher frei wird. Diese Threshold wird vor einer SP Operation oder während einer SP Operationen mit Speicheranforderung überprüft.
VPSEQT	**var.** **(80%)**	Sequential Steal Threshold Dieser Wert drückt den Anteil an Pages eines BP‘s aus, die durch sequentiell verarbeitete Pages belegt sein dürfen. Wird der Grenzwert überschritten, versucht DB2 Pages der Sequential Buffers zu ‚stehlen‘, anstatt auf Pages des Random Buffers zurückzugreifen. Die Einstellung hängt stark von den Art ab, wie Objekte, die diesen Bufferpools zugeordnet sind, verarbeitet werden.
HPSEQT	**var.** **(80%)**	Hiperpool Sequential Steal Threshold (bis Version 8) analog VPSEQT
VPPSEQT	**var.** **(50%)**	Virtual Buffer Pool Parallel Sequential Steal Threshold Drückt den Anteil an Pages aus, die durch Parallelverarbeitung in der VPSEQT-Spannweite belegt werden dürfen.
VPXPSEQT	**var.** **(0%)**	Virtual Buffer Pool Assisting Parallel Sequential Steal Treshold Dies ist der Grenzwert für parallele Operationen, um ein anderes DB2 System in einer Datasharing Group zu unterstützen. Der Wert drückt den Anteil von VPPSEQT aus. Wird der Wert auf 0 gesetzt kann DB2 Sysplex Query Parallelität mit diesem BP nicht unterstützen.
DWQT	**var.** **(50%)**	Deferred Write Threshold Dieser variable Grenzwert legt fest, wieviele Pages durch inhaltliche Änderungen ‚unavailable‘ sein können. Wird eine Grenzwertüberschreitung nach vollzogenem Page Update festgestellt, werden Buffer Writes ausgelöst, die mit einer I/O per Dataset bis zu 128 Pages schreiben, und zwar bis die

Abbrev.	Thres-hold Art	Beschreibung
		Anzahl nicht verfügbarer Pages um mehr als 10% unter den DWQT-Grenzwert fällt. Buffer Writes, durch DWTH verursacht, werden asynchron vollzogen. Die Einstellung variiert mit der Nutzung des Bufferpools: Bei Objekten mit hoher Update-Rate und bei niedrigem DWTH kann es zu häufigen Write Operationen von wenigen veränderten Pages kommen. Write Engines können bis zu 32 Pages, abhängig von der auf DASD zu schreibenden Distanz, auf einmal schreiben. Die max. Spanne der schreibbaren Distanz beträgt einen Zylinder (180 Pages bei 3390) des DASD‘s.
VDWQT	**var.** **(10%)**	Vertical Deferred Write Threshold Analog zu DWQT, jedoch gültig für ein einziges Dataset, das dem BP zugeordnet ist. Dieser Grenzwert wird von DB2 Utilities ignoriert, die eine Übertragungsrate von 64 oder 128 Pages je I/O nutzen (Copy, Load, Reorg, Recover). Buffer Writes, durch VDWTH verursacht, werden asynchron vollzogen. Wird der VDWQT=0 gesetzt, stellt DB2 den Wert implizit auf 1% der Bufferpool-Größe, aber mindestens 40 Pages (bei 4 KB), um synchrone Write Operationen zu vermeiden.

Bufferpool und Performance Tuning

Zur Verbesserung der Performance sollten folgende Aspekte in Erwägung gezogen werden:

- DWQT und VDWQT möglichst niedrig setzen um das Schreiben der veränderten Pages zu forcieren. Dies fördert den Einsatz des asynchronen Writes und reduziert den Aufwand beim Erreichen einer Checkpoint- Intervallgrenze.
- Immer DWQT=25 (dflt. 50) für BP‘s>2000 Pages. Ggf. später auf 20 reduzieren.
  Für BP‘s der DSNDBxx auf 10 setzen.
- VDWQT so nah wie möglich an 128 BP-Pages, aber nicht darunter.
  VDWQT <= DWQT!
- VPSEQT für BP‘s von Objekten ‚accessed randomly‘ den Wert (dflt. 80) reduzieren.
  Bei den BP‘s für DSNDB07 den VPSEQT=95 setzen, aber nie auf 100.
  Bei CPU- oder I/O-Parallelität den VPPSEQT entspr. anpassen, sonst irrelevant.
- LOB's über separate BP's abwickeln, DWQT=0 setzten um massive Write Operationen bei einem Commit oder Checkpoint zu verhindern.
- Die Over-Allocation von DBM1 über die vorhandene Speicherkapazität hinaus (‚page-ins‘), führt zu Paging und (schweren) Performance-Einbußen.
- Automatic Bufferpool Management mit AUTOSIZE(YES) empfiehlt sich besonders bei nicht kritischen Systemen, z.B. Test oder Entwicklung.
- Optimierung von Sort-Vorgängen:
  Sortpool vergrößern, separate Devices für Work Files, mehr und größere Work Files, separater Bufferpool für Work File Database. Ein VPSEQT von 99% verhindert, dass Spacemap Pages, die direkt gelesen werden, durch massive Prefetch Operationen überschrieben werden. VPSEQT kann auf 100% gesetzt werden, wenn keine Sparse Indexes genutzt werden um auf Work Files zuzugreifen. PRIMARY_ACCESSTYPE = T zeigf an, dass DB2 dnyamisch einen s.g. Sparse Index (Behelfsindex) für die Inner Table erzeugt, um die Work File zu durchsuchen, die für die Inner Table erzeugt wurde. DWQT und/oder VDWQT hochsetzen.
- Unter Umständen bietet sich an, je nach Verarbeitung (online oder batch), die Thresholds dynamisch zu verändern. Beispielsweise Dynamic Prefetch Operationen in der Nacht zu bevorzugen, am Tag online jedoch Random Access den Vorrang zu geben.