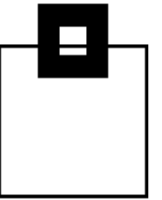


BUFFERPOOL Tuning The Next Generation

Roy Boxwell,
SEGUS Inc.



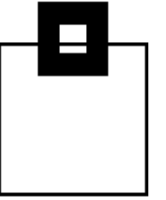
Agenda



- What use are BUFFERPOOLS?
- Is it worth tuning?
- How do you tune them?
- What about GROUP BUFFERPOOLS?
- The modern way to visualize BP/GBP problems
- Q&A



Agenda

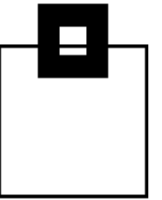
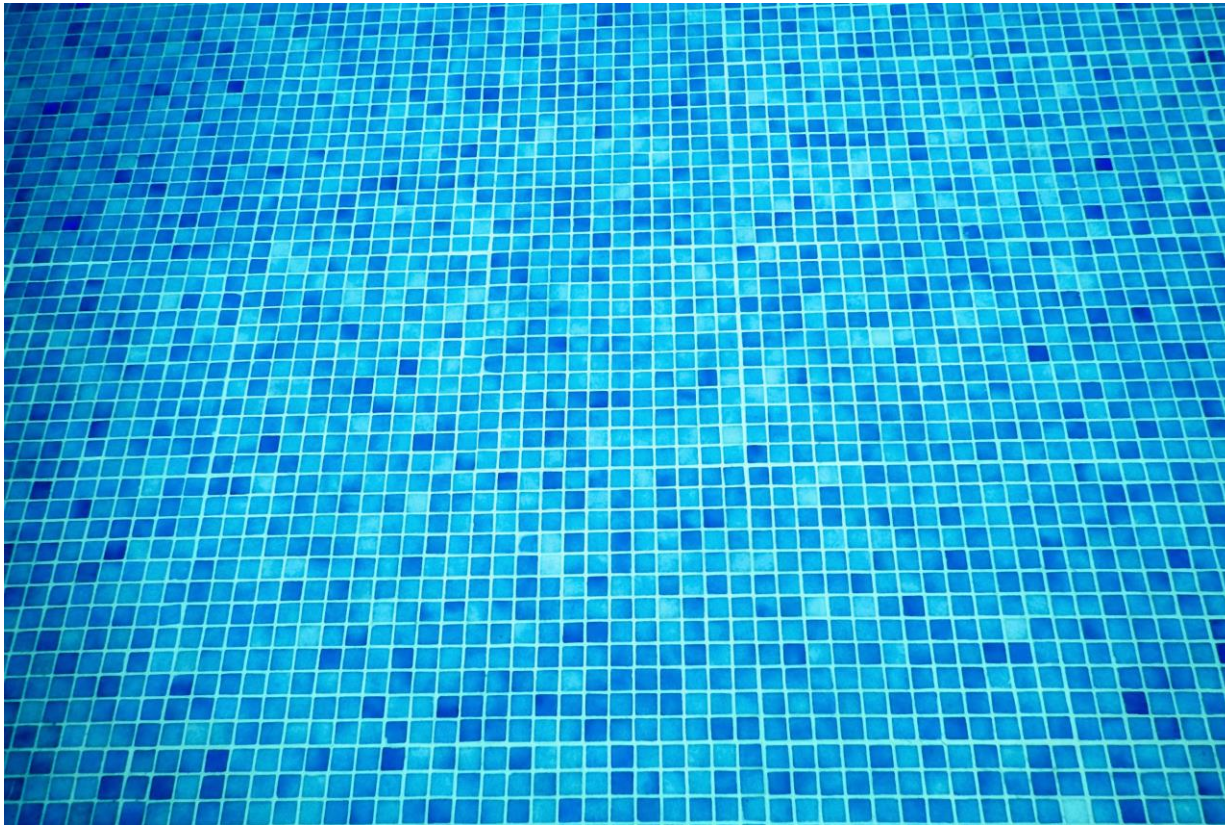


- **What use are BUFFERPOOLS?**
- Is it worth tuning?
- How do you tune them?
- What about GROUP BUFFERPOOLS?
- The modern way to visualize BP/GBP problems
- Q&A



What use are BUFFERPOOLS?

Bufferpools:



What use are BUFFERPOOLS?

Real bufferpools (BPs) have existed since the get-go of DB2 (when the B was big!)

The idea, back in the day, was to have two sizes of BP for matching the two sizes of DB2 pages - namely 4KB and 32KB.

DB2 started *very* small - we only had FOUR pools! Three 4KB and a single 32KB:

Buffer Pool Calculation	XA Default	370 Default
Buffers for BP0 ___ * 4K = ___	224 * 4K = 896K	56 * 4K = 224K
Buffers for BP1 + ___ * 4K = ___	+ 0 * 4K = 0K	0 * 4K = 0K
Buffers for BP2 + ___ * 4K = ___	+ 0 * 4K = 0K	0 * 4K = 0K
Buffers for BP32K + ___ * 32K = ___	+12 * 32K = 384K	3 * 32K = 96K
	<hr/> 1280K	<hr/> 320K

Figure 29. Buffer Pool Size Calculation

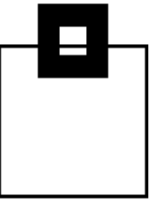
What use are BUFFERPOOLS?

In DB2 V3.1 we got the boost of BP0 to BP49 and BP32K, BP32K1 to BP32K9. BP0 had 2,000 pages as a default and BP32K only 24(!) pages as a default...

In DB2 V6.1 IBM introduced the zero buffers... BP8K0 – BP8K9 and BP16K0 – BP16K9 thus endearing themselves into the heart of all future DBAs!

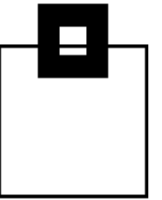
In DB2 V8.1 BP0 default jumped to 20,000 pages, BP8K0 was 1,000 pages, BP16K0 was 500 pages and BP32K was 250 pages

In Db2 11 the B went lower case...



What use are BUFFERPOOLS?

In fact, the defaults have not changed at all so they must be good and stable, yes?



What use are BUFFERPOOLS?

In fact, the defaults have not changed at all so they must be good and stable, yes?

```
DSNTIP1          INSTALL DB2 - BUFFER POOL SIZES - PANEL 1
====>

Enter 4 KB buffer pool sizes in number of pages.
 1 BP0  ==> 20000      18 BP17 ==> 0         35 BP34 ==> 0
 2 BP1  ==> 0          19 BP18 ==> 0         36 BP35 ==> 0
 3 BP2  ==> 0          20 BP19 ==> 0         37 BP36 ==> 0
 4 BP3  ==> 0          21 BP20 ==> 0         38 BP37 ==> 0
 5 BP4  ==> 0          22 BP21 ==> 0         39 BP38 ==> 0
 6 BP5  ==> 0          23 BP22 ==> 0         40 BP39 ==> 0
 7 BP6  ==> 0          24 BP23 ==> 0         41 BP40 ==> 0
 8 BP7  ==> 0          25 BP24 ==> 0         42 BP41 ==> 0
 9 BP8  ==> 0          26 BP25 ==> 0         43 BP42 ==> 0
10 BP9  ==> 0          27 BP26 ==> 0         44 BP43 ==> 0
11 BP10 ==> 0          28 BP27 ==> 0         45 BP44 ==> 0
12 BP11 ==> 0          29 BP28 ==> 0         46 BP45 ==> 0
13 BP12 ==> 0          30 BP29 ==> 0         47 BP46 ==> 0
14 BP13 ==> 0          31 BP30 ==> 0         48 BP47 ==> 0
15 BP14 ==> 0          32 BP31 ==> 0         49 BP48 ==> 0
16 BP15 ==> 0          33 BP32 ==> 0         50 BP49 ==> 0
17 BP16 ==> 0          34 BP33 ==> 0

PRESS:  ENTER to continue  RETURN to exit  HELP for more information
```


What use are BUFFERPOOLS?

In fact, the defaults have not changed at all so they must be good and stable, yes?

```
DSNTIP1  
====>
```

```
INSTALL DB2 - BUFFER POOL SIZES - PANEL 1
```

```
Enter 4 KB buffer pool sizes in number of pages.  
1 BP0 ==> 20000  
2 BP1 ==> 0  
3 BP2 ==> 0  
4 BP3 ==> 0  
5 BP4 ==> 0  
6 BP5 ==> 0  
7 BP6 ==> 0  
8 BP7 ==> 0  
9 BP8 ==> 0  
10 BP9 ==> 0  
11 BP10 ==> 0  
12 BP11 ==> 0  
13 BP12 ==> 0  
14 BP13 ==> 0  
15 BP14 ==> 0  
16 BP15 ==> 0  
17 BP16 ==> 0
```

```
PRESS: ENTER to continue
```

```
DSNTIP2  
====>
```

```
INSTALL DB2 - BUFFER POOL SIZES - PANEL 2
```

```
Enter 8 KB, 16KB, and 32 KB buffer pool sizes in number of pages.
```

```
1 BP8K0 ==> 2000      11 BP16K0 ==> 500      21 BP32K  ==> 250  
2 BP8K1 ==> 0        12 BP16K1 ==> 0        22 BP32K1 ==> 0  
3 BP8K2 ==> 0        13 BP16K2 ==> 0        23 BP32K2 ==> 0  
4 BP8K3 ==> 0        14 BP16K3 ==> 0        24 BP32K3 ==> 0  
5 BP8K4 ==> 0        15 BP16K4 ==> 0        25 BP32K4 ==> 0  
6 BP8K5 ==> 0        16 BP16K5 ==> 0        26 BP32K5 ==> 0  
7 BP8K6 ==> 0        17 BP16K6 ==> 0        27 BP32K6 ==> 0  
8 BP8K7 ==> 0        18 BP16K7 ==> 0        28 BP32K7 ==> 0  
9 BP8K8 ==> 0        19 BP16K8 ==> 0        29 BP32K8 ==> 0  
10 BP8K9 ==> 0       20 BP16K9 ==> 0        30 BP32K9 ==> 0
```

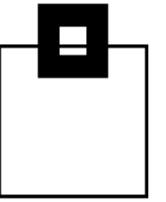
```
31 DEFAULT 4-KB BUFFER POOL FOR USER DATA ==> BP1      BP0      - BP49  
32 DEFAULT 8-KB BUFFER POOL FOR USER DATA ==> BP8K0    BP8K0    - BP8K9  
33 DEFAULT 16-KB BUFFER POOL FOR USER DATA ==> BP16K0   BP16K0   - BP16K9  
34 DEFAULT 32-KB BUFFER POOL FOR USER DATA ==> BP32K    BP32K    - BP32K9  
35 DEFAULT BUFFER POOL FOR USER LOB DATA  ==> BP0      BP0      - BP32K9  
36 DEFAULT BUFFER POOL FOR USER XML DATA  ==> BP16K0   BP16K0   - BP16K9  
37 DEFAULT BUFFER POOL FOR USER INDEXES    ==> BP0      BP0      - BP32K9  
PRESS: ENTER to continue RETURN to exit HELP for more information
```

What use are BUFFERPOOLS?

Ok, so we have a nice set of default sizes but what are they used for?

Well, even today on our “fake” DASD there is a nasty thing called I/O and I/O is slow!

The best I/O does not cause a disk seek at all, in fact that is the entire point of a BP. A piece of required data is found in memory so that the data is instantly available to the application process.



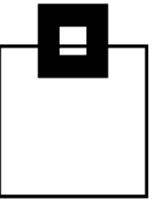
What use are BUFFERPOOLS?

Why do we have so many BPs in Db2 for z/OS? Why not just one huge area of RAM stuffed full of data?

Well, the answer to that is “Horses for Courses”.

The performance of any given BP is strongly related to the applications running and using it. Think of a process that is sequentially reading through a table for summation purposes. It reads data but will **never want to read it again.**

Is this “good” for the BP?



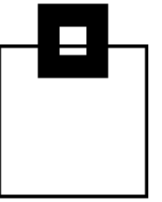
What use are BUFFERPOOLS?

Why do we have so many BPs in Db2 for z/OS? Why not just one huge area of RAM stuffed full of data?

Well, the answer to that is “Horses for Courses”.

The performance of any given BP is strongly related to the applications running and using it. Think of a process that is sequentially reading through a table for summation purposes. It reads data but will **never want to read it again.**

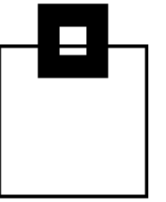
Is this “good” for the BP? Nope.



What use are BUFFERPOOLS?

Now imagine an application process that is randomly reading data through an index. It fetches the leaf and non-leaf pages into the BP as it needs them and then carries on.

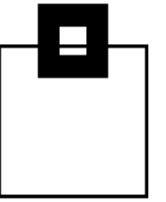
When it needs that “used” leaf page it will find it again in the BP.
Is this a “good” use of the BP?



What use are BUFFERPOOLS?

Now imagine an application process that is randomly reading data through an index. It fetches the leaf and non-leaf pages into the BP as it needs them and then carries on.

When it needs that “used” leaf page it will find it again in the BP. Is this a “good” use of the BP? Yes.

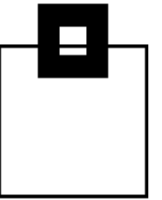


What use are BUFFERPOOLS?

Now imagine an application process that is randomly reading data through an index. It fetches the leaf and non-leaf pages into the BP as it needs them and then carries on.

When it needs that “used” leaf page it will find it again in the BP. Is this a “good” use of the BP? Yes.

If both of these applications share the same BP this is obviously not good, but this is what most, if not all, Db2 shops do!

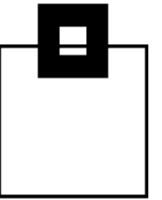


What use are BUFFERPOOLS?

Now imagine a sort – Yes I am talking about DSNDB07 usage here! You might not know it but SORT requires a BP as well. What are the odds of a repeated reread in a sort pool?

You can imagine they are pretty low!

Sort should **always** be in its own little/large pool with VPSEQT set to 99%.

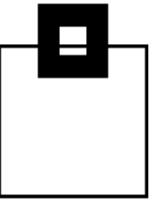


What use are BUFFERPOOLS?

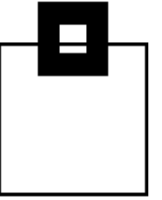
Now imagine a sort – Yes I am talking about DSNDB07 usage here! You might not know it but SORT requires a BP as well. What are the odds of a repeated reread in a sort pool?

You can imagine they are pretty low!

Sort should **always** be in its own little/large pool with VPSEQT set to 99%. Unless you are not using WFDBSEP=YES, as then your DGTTS need random access. Plus, if you are using Sparse Index access it also requires random access. The recommendation then is to set VPSEQT 90%, DWQT 50% and VDWQT 10%. Monitor synchronous I/O's and if low to zero raise VPSEQT.



What use are BUFFERPOOLS?



What about LRU, FIFO, NONE when stealing pages?

“It depends” raises its ugly head here!

LRU is least recently used - so that the “stalest” pages can be got rid of and replaced with newer ones and is, naturally, the default.



NONE is great for “permanent in-memory data” Xref tables etc.



FIFO is great if you really do not care about the least recently used logic. It is really a niche use case though.

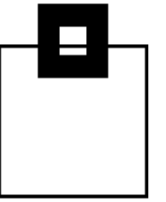


What use are BUFFERPOOLS?

Most shops have no personnel to look at, monitor, tune and change the BPs at their site. The expertise is “graying” and lots of people are “afraid” of changing something so crucial as a BP.

This is not healthy!

BP usage always changes over time! (Death by cut-and-paste is a classic...)

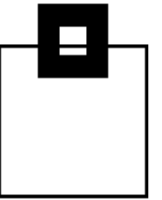


What use are BUFFERPOOLS?

The Db2 Directory and Catalog are the most important part of any Db2 system. They are the meta-data repositories of the entire system and contain **everything** you need to run, but these objects go into only the listed default BPs and I will bet that most shops also use these BPs for application data – This I call BP Pollution.

The first thing you must do is move all non-Db2 Directory and Catalog objects **out** of BP0, BP8K0, BP16K0 and BP32K.

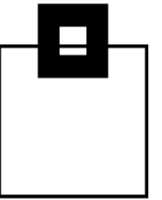
This can be tough, but it must be done! It is useless starting to tune BPs when their usage is completely broken!



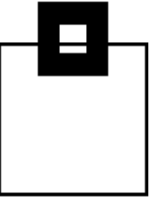
What use are BUFFERPOOLS?

A standard set of “Rules of Thumb” is:

- 1) The Db2 Directory and Catalog on their own
- 2) Application tablespaces and indexes kept apart
- 3) LOB and XML on their own
- 4) Sort on its own (use BP7 and BP32K7 here)
- 5) In-memory tables (PGSTEAL(NONE) on their own)
- 6) Randomly accessed data on their own
- 7) Sequentially accessed data on their own
- 8) GBP correctly sized
- 9) The rest...



Agenda

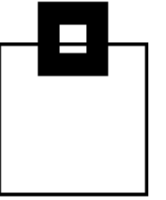


- What use are BUFFERPOOLS?
- Is it worth tuning?
- How do you tune them?
- What about GROUP BUFFERPOOLS?
- The modern way to visualize BP/GBP problems
- Q&A



Is it worth tuning?

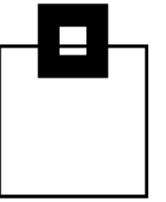
The answer is a massive



Is it worth tuning?

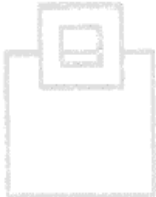
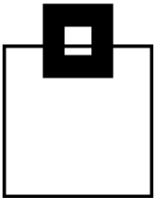
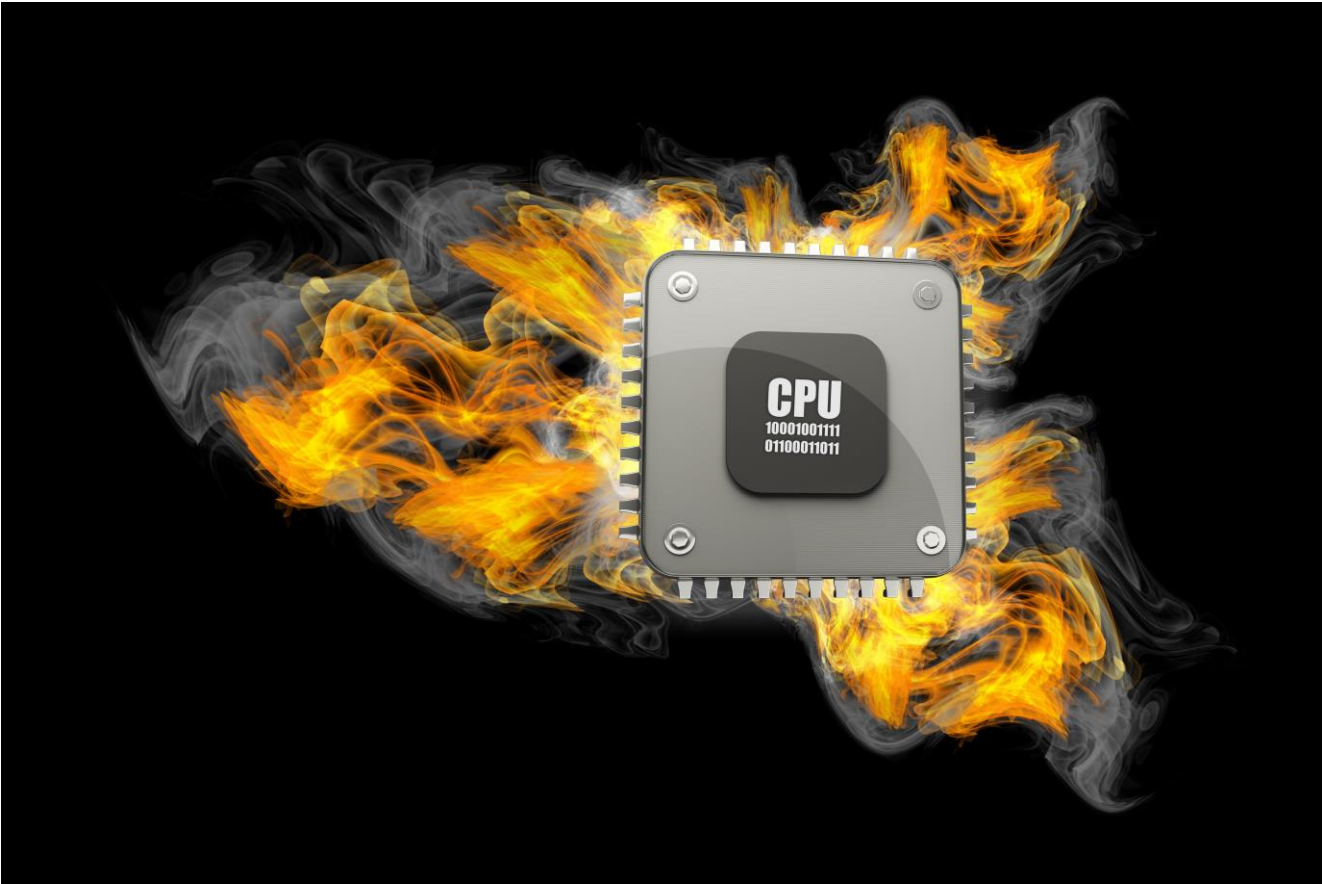
The answer is a massive

YES!



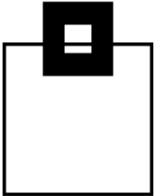
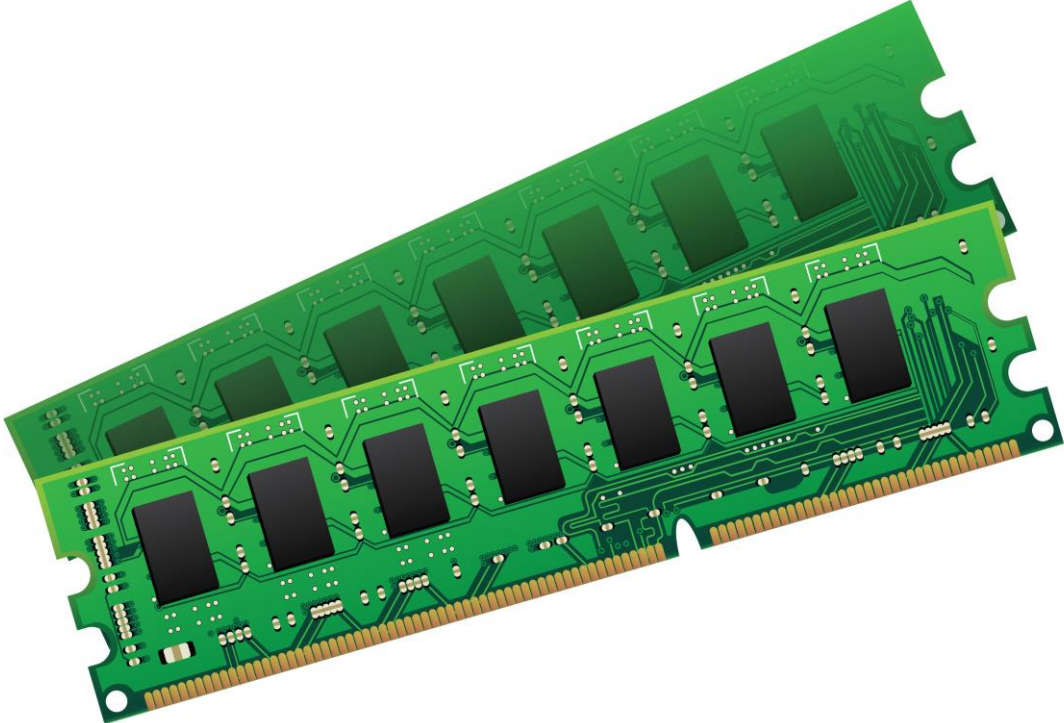
Is it worth tuning?

Your CPU is on fire!



Is it worth tuning?

Your
RAM is
choked?



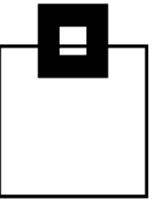
Is it worth tuning?

It is often written that the quickest and best results of any type of tuning are indeed with BP and GBPs.

SQL Tuning is still, obviously, required but the big system-wide ROI can be had in the BP and GBP area.

IBM state that “no brainer” options such as PGFIX(YES) give up to 8% cpu savings. That is at the *system* level!

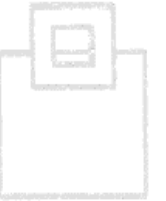
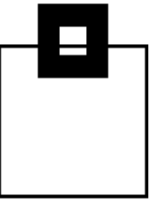
Using large frame sizes (1Mb or 2GB) saves up to 4%.



Is it worth tuning?

If you can imagine a system where it is actively paging, actively reading and rereading data and index pages all the time just because it cannot find the data in the BP it is clear that you can

“Tune the SQL until you die, it will not get faster!”



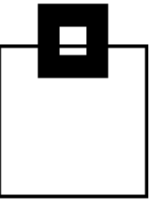
Is it worth tuning?

Another very popular problem is the:

“Stuff it in BP8K0, I know that BP exists!”

This is especially popular for COMPRESS YES indexes – Thus solving one problem and introducing another, even worse, problem at the same time!

Naturally, the BP problem is not seen and everyone wonders why performance tanks every now and again...



Is it worth tuning?

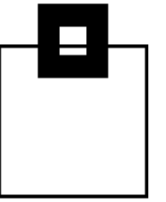
Another very popular problem is the:

“Stuff it in BP8K0, I know that BP exists!”

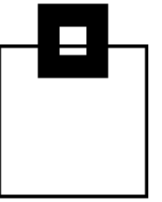
This is especially popular for COMPRESS YES indexes – Thus solving one problem and introducing another, even worse, problem at the same time!

Naturally, the BP problem is not seen and everyone wonders why performance tanks every now and again...

Death by sync I/O is not a pretty sight...



Agenda

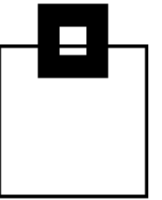


- What use are BUFFERPOOLS?
- Is it worth tuning?
- **How do you tune them?**
- What about GROUP BUFFERPOOLS?
- The modern way to visualize BP/GBP problems
- Q&A



How do you tune them?

There is an ALTER command...

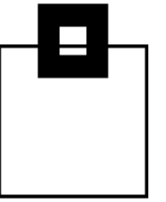


How do you tune them?

There is an ALTER command...

Now in Db2 for LUW you would just throw a few more CPUs or GBs of RAM at the problem and walk away a happy bunny.

On z/OS it is a little bit more tricky but I know of shops that have basically done the same!



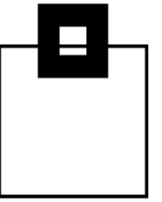
How do you tune them?

There is an ALTER command...

Now in Db2 for LUW you would just throw a few more CPUs or GBs of RAM at the problem and walk away a happy bunny.

On z/OS it is a little bit more tricky but I know of shops that have basically done the same!

First, you must find the current state of your BPs.



How do you tune them?

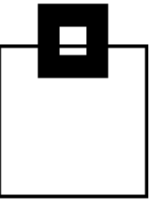
There is an ALTER command...

Now in Db2 for LUW you would just throw a few more CPUs or GBs of RAM at the problem and walk away a happy bunny.

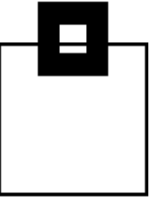
On z/OS it is a little bit more tricky but I know of shops that have basically done the same!

First, you must find the current state of your BPs.

You must then ALTER them to do what they should be doing!



Agenda



- What use are BUFFERPOOLS?
- Is it worth tuning?
- How do you tune them?
- **What about GROUP BUFFERPOOLS?**
- The modern way to visualize BP/GBP problems
- Q&A

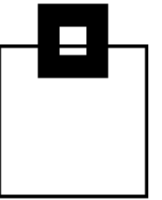


What about GROUP BUFFERPOOLS?

These are the forgotten “zombies” of the Db2 for z/OS BP world!

They are incredibly important for the well-running of any data-sharing system and basically require the same style of tuning as normal – local – BPs.

They have sizes and thresholds just like local BPs but they have normally been completely forgotten about in most shops!



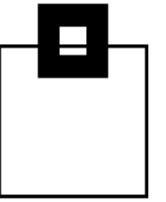
What about GROUP BUFFERPOOLS?

These are the forgotten “zombies” of the Db2 for z/OS BP world!

They are incredibly important for the well-running of any data-sharing system and basically require the same style of tuning as normal – local – BPs.

They have sizes and thresholds just like local BPs but they have normally been completely forgotten about in most shops!

First, you must find the current state of your Group BPs.



What about GROUP BUFFERPOOLS?

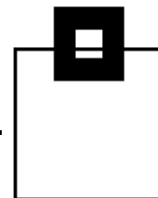
These are the forgotten “zombies” of the Db2 for z/OS BP world!

They are incredibly important for the well-running of any data-sharing system and basically require the same style of tuning as normal – local – BPs.

They have sizes and thresholds just like local BPs but they have normally been completely forgotten about in most shops!

First, you must find the current state of your Group BPs.

You must then ALTER them to do what they should be doing!

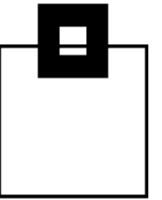


What about GROUP BUFFERPOOLS?

Naturally it is not always just a simple ALTER command that is required, but a change of the CFRM Policy to increase the INITSIZE as these are very often way too small!

In fact, if you change the INITSIZE, GBPCACHE or RATIO the ALTER has no instant affect and will require a rebuild:

SETXCF START,REBUILD



What about GROUP BUFFERPOOLS?

If you are duplexed, and I sincerely hope you all are, you must first go to simplex mode:

```
SETXCF STOP, RB, DUPLEX, STRNAME=XXXXXXXXX.YYYYYY, KEEP=OLD
```

Then issue the:

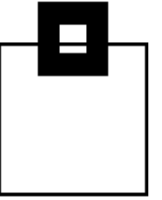
```
SETXCF START, REBUILD
```

And then, if not going to GBPCACHE(NO), go back to duplex mode:

```
SETXCF START, RB, DUPLEX, STRNAME=XXXXXXXXX.YYYYYY
```

Where xxxxxxxx is Group Name and yyyyyy is the Group BP Name.

Agenda



- What use are BUFFERPOOLS?
- Is it worth tuning?
- How do you tune them?
- What about GROUP BUFFERPOOLS?
- **The modern way to visualize BP/GBP problems**
- Q&A

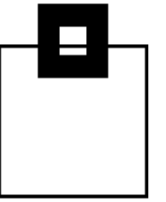


The modern way to visualize BP/GBP problems

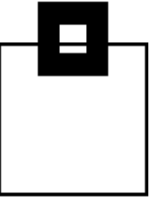
We now know that we have a problem!

How can we actually visualize this and do stuff?

The archaic way that 3270 green screen outputs data and/or the extremely detailed obscure formulae that must be used and/or the different sources of data that must be trawled, all make it “non-trivial” ...



The modern way to visualize BP/GBP problems



We now know that we have a problem!

How can we actually visualize this and do stuff?

The archaic way that 3270 green screen outputs data and/or the extremely detailed obscure formulae that must be used and/or the different sources of data that must be trawled, all make it “non-trivial”...

How “non-trivial”???



The modern way to visualize BP/GBP problems

You have to get the correct info from 100's of metrics to then calculate and check all the performance-relevant thresholds:

System residency

Random residency

Sequential residency

System Hit Ratio

Application Hit Ratio

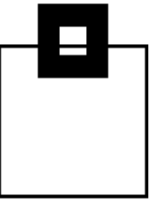
DMTH Threshold hit

Prefetch disabled no buffer

Prefetch disabled no read engine

DWQT hit rate / Second

VDWQT hit rate / Second



The modern way to visualize BP/GBP problems

You have to get the correct info from 100's of metrics to then calculate and check all the performance-relevant thresholds:

Bufferpool too big

Large Bufferpool VDWQT

VPSEQT should be changed

Random I/Os per second

Prefetch size

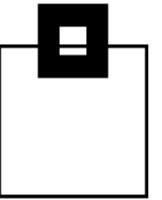
No. of page updates for each page written

No. of pages written for each write I/O

Page arrival rate

I/O Intensity

Bufferpool Intensity



The modern way to visualize BP/GBP problems

You have to get the correct info from 100's of metrics to then calculate and check all the performance-relevant thresholds:

Getpage rate

Frame boundary

Frame sizing

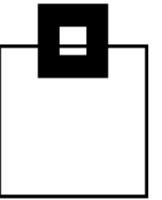
Frame size with LFAREA

PGFIX(NO) used

Page-ins for read required

Page-ins for write required Random Sync I/O

Bufferpool paging



The modern way to visualize BP/GBP problems

You have to get the correct info from 100's of metrics to then calculate and check all the performance-relevant thresholds:

GBP INITSIZE

GBP Writes failed due to lack of storage

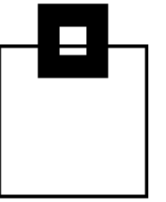
GBP Sync read XI miss ratio – high

GBP Sync read XI data not returned per day – high

GBP Sync read XI data not returned per second – high

GBP Cross Invalidations (XI) due to directory reclaims

GBP Castout



The modern way to visualize BP/GBP problems

You have to get the correct info from 100's of metrics to then calculate and check all the performance-relevant thresholds:

GBP Reclaims for directory entries

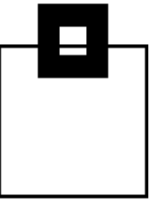
GBP Snapshot

GBP Hit ratio

GBP Negotiations for Spacemap

GBP Negotiations for Datapages

GBP Duplexing

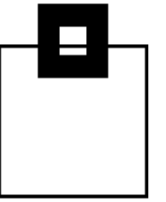


The modern way to visualize BP/GBP problems

So if your
high speed
data looks
like this...



www.123rf.com



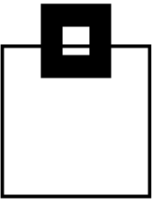
The modern way to visualize BP/GBP problems

So if your
high speed
data looks
like this...

Oh dear!!!



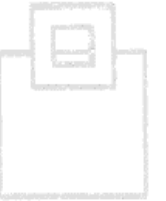
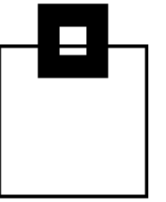
www.123rf.com



The modern way to visualize BP/GBP problems

Not only do you have to gather and compute all of your values, but you have to know which threshold and/or which value to ALTER to actually do the corrective action!

This is all pretty nasty work that someone probably did way back in the 1990's but since then it has not been updated...

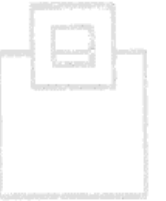
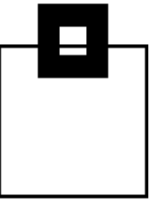


The modern way to visualize BP/GBP problems

Not only do you have to gather and compute all of your values, but you have to know which threshold and/or which value to ALTER to actually do the corrective action!

This is all pretty nasty work that someone probably did way back in the 1990's but since then it has not been updated...

Let WorkloadExpert™ for Db2 z/OS (WLX) do the heavy lifting!



The modern way to visualize BP/GBP problems

The screenshot shows the SQL WorkloadExpert for Db2 z/OS application window. The title bar reads "SQL WorkloadExpert for Db2 z/OS" and "Selected profile: SC10-IQA0610 - IQA0610 - Z100SC10 - 192.168.9.98 - 5125". The main menu bar includes "Application development", "Database administration", "Workload / Performance optimization & tuning", "Audit", "Error handling", and "Other". A red callout box points to the "Workload / Performance optimization & tuning" menu item.

W LX comes with more than 70 prepared Use Cases, categorized into six areas of interest for quick and easy workload analysis scenarios.

The bottom of the screenshot shows a taskbar with various icons and a system tray displaying the time "9:12 AM" and date "8/1/23".

The modern way to visualize BP/GBP problems

The screenshot shows the SQL WorkloadExpert for Db2 z/OS application window. The title bar reads "SQL WorkloadExpert for Db2 z/OS" and includes a "Selected profile" field with the value "SC10-IQA0610 - IQA0610 - Z100SC10 - 192.168.9.98 - 5125". The main menu bar contains "Application development", "Database administration", "Workload / Performance optimization & tuning", "Audit", "Error handling", and "Other". The "Workload / Performance optimization & tuning" menu is open, listing several options: "CPU intensive SQLs", "CPU usage of packages", "Delay detection", "Disk I/O", "DSC/SSC flush rates", "PKLIST problem detection", "SELECT only detection", "Up and Down scaling", and "WLX KPIs and summaries". A red callout box with a red border and a red arrow pointing to "CPU intensive SQLs" contains the text: "Here is an example about finding the most expensive SQL queries." The system tray at the bottom shows the time as 9:13 AM on 8/1/23.

The modern way to visualize BP/GBP problems

SQL WorkloadExpert for Db2 z/OS

Selected profile: DD10SEC-IQA061QB - IQA061QB - Z100DD10 - 192.168.9.98 - 15151

Application Workload

Projection Selection Sorting Preferences Favorites

Label	Description	Add
PROGRAM	Package Name	>
PACKAGE_COLLID	Collection ID	>
STMT_ORIGIN	Statement Origin	>
NUMBER_OF_STATEMENTS	Number of Statements	>
TOTAL_CPU_TIME	Sum of CPU Time	>
HIGHEST_CPU_TIME	Highest CPU Time	>
TOTAL_ELAPSED_TIME	Sum of Elapsed Time	>
HIGHEST_ELAPSED_TIME	Highest Elapsed Time	>
TOTAL_GETPAGES	Sum of GETPAGES	>
HIGHEST_GETPAGES	Highest GETPAGES	>
TOTAL_SYNC_BUFFER_READS	Sum of Synchronous Buffer Reads	>
TOTAL_SYNC_BUFFER_WRITES	Sum of Synchronous Buffer Writes	>
TOTAL_ROWS_EXAMINED	Sum of Rows examined	>
TOTAL_ROWS_PROCESSED	Sum of Rows processed	>
TOTAL_INDEX_SCANS	Sum of Index scans	>

Label	Description	Remove
PRIM_AUTHOR	Primary Authorization ID	x
TOTAL_EXECUTIONS	Sum of Executions	x
AVERAGE_CPU_TIME	Average CPU Time	x
AVERAGE_ELAPSED_TIME	Average Elapsed Time	x
AVERAGE_GETPAGES	Average GETPAGES	x

Cancel OK

9:42 AM 8/1/23

Each Use Case guides you through customization to fit exactly your needs.

First, we select the data we want to report (Projection)...

The modern way to visualize BP/GBP problems

The screenshot shows the SQL WorkloadExpert for Db2 z/OS interface. The main window title is "SQL WorkloadExpert for Db2 z/OS" and the selected profile is "DD10SEC-IQA061QB - IQA061QB - Z100DD10 - 192.168.9.98 - 15151". The "Application Workload" section is active, and the "Selection" tab is selected. The "Available Columns" list includes various performance metrics like AVERAGE_CPU_TIME, AVERAGE_ELAPSED_TIME, etc. The "Selected Columns" list shows "WLX_TIMESTAMP" with an operation of "=" and a value of "newest", and "STMT_ORIGIN" with an operation of "=" and a value of "D". A red callout box highlights the text: "... then you can apply some filtering for the result set of interest (Selection) ...".

Label	Description	Add
AVERAGE_CPU_TIME	Average CPU Time	>
AVERAGE_ELAPSED_TIME	Average Elapsed Time	>
AVERAGE_GETPAGES	Average GETPAGES	>
PACKAGE_COLLID	Collection ID	>
CPU_TIME	CPU Time	>
CURRENT_DEGREE_SW	Current Degree special register	>
CUR_PRECISION_SW	Current Precision special register	>
CURRENT_RULES_SW	Current Rules special register	>
CUR_SQLID	Current SQL ID	>
CURRENT_DATA_SW	CURRENTDATA option	>
CURSOR_HOLD_SW	Cursor prepared with hold indicator	>
DYNAMIC_RULE_SW	DYNAMICRULES option	>
ELAPSE_TIME	Elapsed Time	>
END_USERID	End User ID	>
EVECTIONS	Evections	>

Label	Operation	Value	Description	Remove
WLX_TIMESTAMP	=	newest	WLX Key	x
STMT_ORIGIN	=	D	Statement Origin	x

The modern way to visualize BP/GBP problems

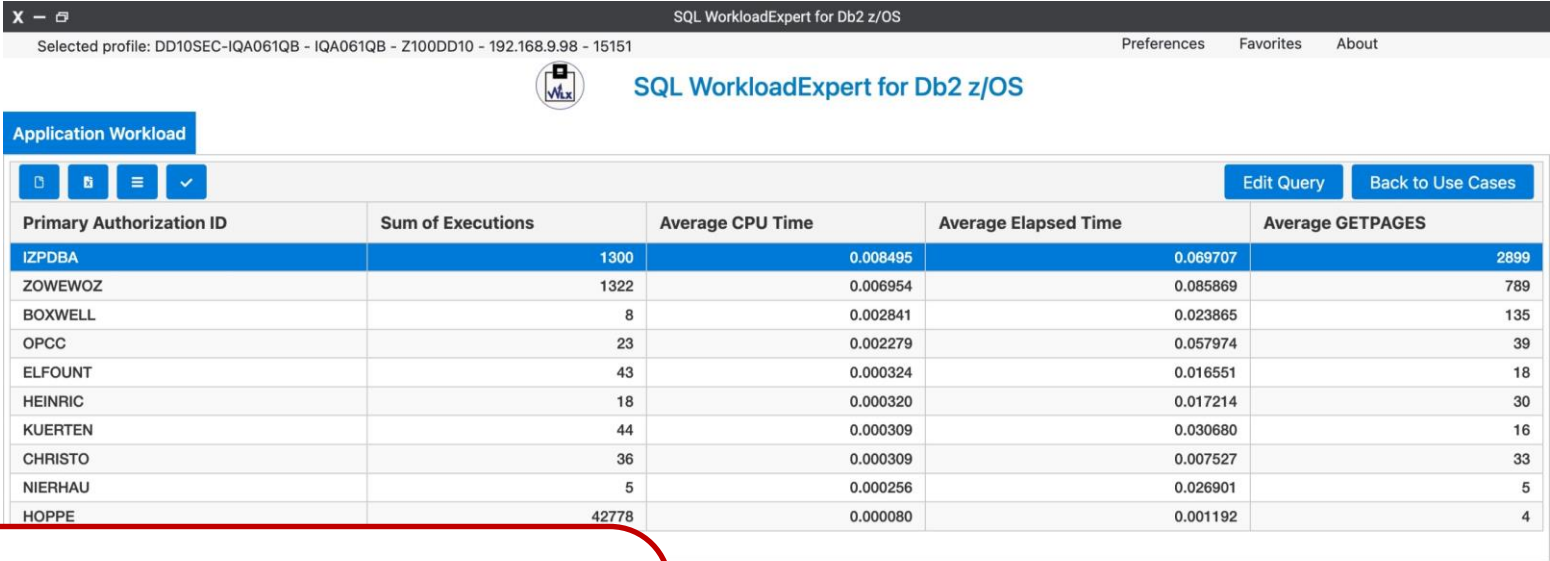
The screenshot shows the SQL WorkloadExpert for Db2 z/OS application. The main window title is "SQL WorkloadExpert for Db2 z/OS" and the selected profile is "DD10SEC-IQA061QB - IQA061QB - Z100DD10 - 192.168.9.98 - 15151". The "Application Workload" section is active, and the "Sorting" tab is selected. The "Available Columns" table lists various performance metrics, and the "Selected Columns" table shows "AVERAGE_CPU_TIME" selected with a "Direction" dropdown set to "DE...". A red-bordered callout box highlights the sorting options.

Label	Description	Add
AVERAGE_ELAPSED_TIME	Average Elapsed Time	>
AVERAGE_GETPAGES	Average GETPAGES	>
PACKAGE_COLLID	Collection ID	>
HIGHEST_CPU_TIME	Highest CPU Time	>
HIGHEST_ELAPSED_TIME	Highest Elapsed Time	>
HIGHEST_GETPAGES	Highest GETPAGES	>
NUMBER_OF_STATEMENTS	Number of Statements	>
PROGRAM	Package	>
PRIM_AUTHOR	Primary Authorization ID	>
STMT_ORIGIN	Statement Origin	>
TOTAL_CPU_COST	Sum of CPU costs	>
TOTAL_CPU_TIME	Sum of CPU Time	>
TOTAL_ELAPSED_TIME	Sum of Elapsed Time	>
TOTAL_EXECUTIONS	Sum of Executions	>
TOTAL_GETPAGES	Sum of GETPAGES	>

Label	Description	Direction	Remove
AVERAGE_CPU_TIME	Average CPU Time	DE... v	x

... and you can set the order of the result set (Sorting).
Any of this can be kept and/or shared for later execution, or even for automatic reporting.

The modern way to visualize BP/GBP problems

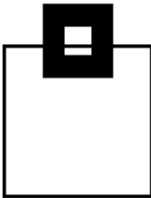


The screenshot shows the SQL WorkloadExpert for Db2 z/OS interface. The title bar indicates the selected profile: DD10SEC-IQA061QB - IQA061QB - Z100DD10 - 192.168.9.98 - 15151. The main content area is titled "Application Workload" and contains a table with the following data:

Primary Authorization ID	Sum of Executions	Average CPU Time	Average Elapsed Time	Average GETPAGES
IZPDBA	1300	0.008495	0.069707	2899
ZOWEWOZ	1322	0.006954	0.085869	789
BOXWELL	8	0.002841	0.023865	135
OPCC	23	0.002279	0.057974	39
ELFOUNT	43	0.000324	0.016551	18
HEINRIC	18	0.000320	0.017214	30
KUERTEN	44	0.000309	0.030680	16
CHRISTO	36	0.000309	0.007527	33
NIERHAU	5	0.000256	0.026901	5
HOPPE	42778	0.000080	0.001192	4

The result set can be used for further in-depth drill-down analysis, cross-reference reporting, exporting into a pdf/excel, or ...

The modern way to visualize BP/GBP problems



The screenshot shows the SQL WorkloadExpert for Db2 z/OS interface. A 'Create Report' dialog box is open, allowing configuration of a report. The dialog includes fields for Profile, Category Column, Value Columns, and Chart Type. The 'Chart Type' section has radio buttons for Horizontal bar, Vertical bar, Pie, Line, and Radar, with 'Radar' selected. A 'Create Report' button is at the bottom of the dialog. In the background, a table displays workload data.

Primary Authorization ID	Sum of C	Examined	Sum of Rows processed
IZPDBA		11421000	2636900
		3751245	118815
		121433	287297
		1286	943
		11931	990
		2847	83
		4035	159
		3297	340
		146	149
		23	0

... to generate a graphical report of many kinds (bars, pies, lines, or radar charts ...)



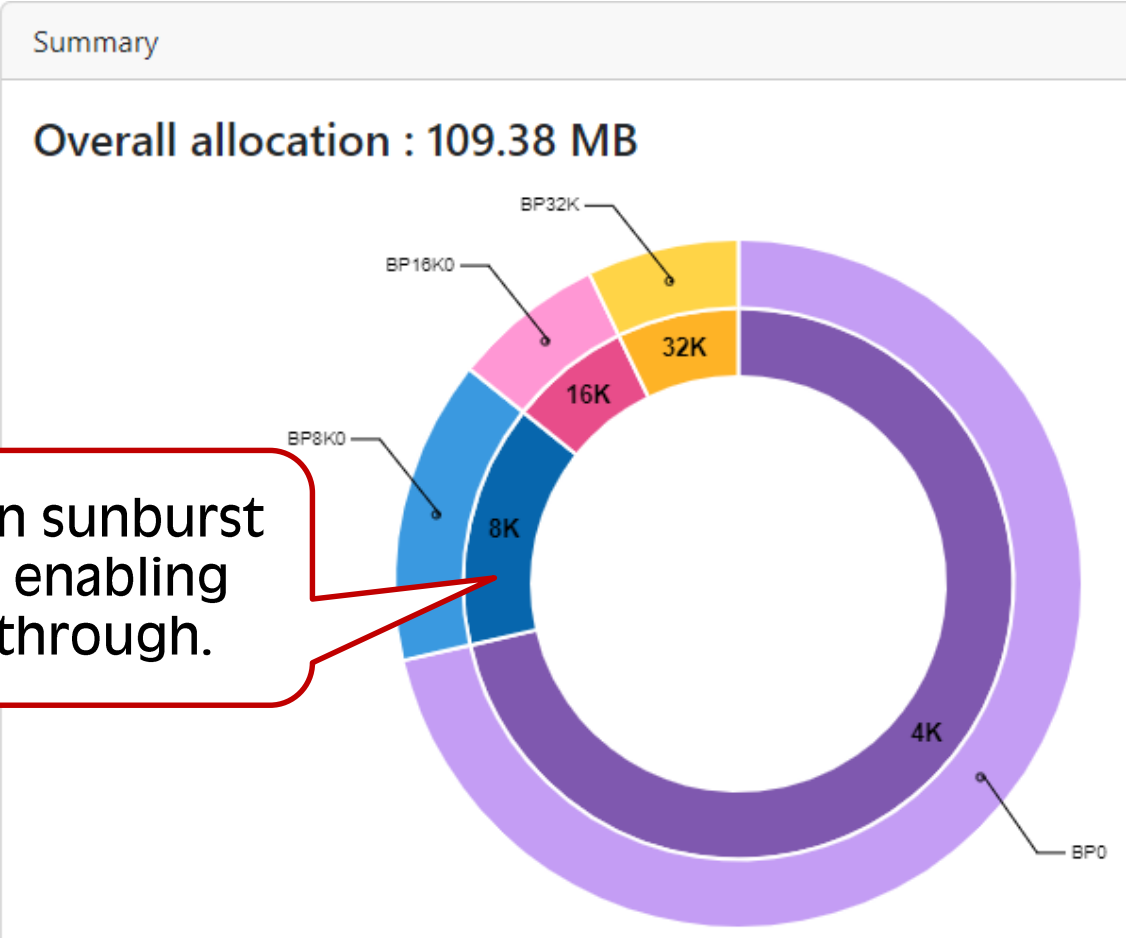
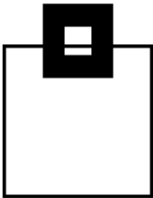
The modern way to visualize BP/GBP problems

The screenshot shows the 'Space AssuranceExpert for Db2 z/OS' application interface. The main content area is titled 'Preferences' and displays the following settings:

- Profile:** SC10-MVNXTEST (selected in a dropdown menu)
- Profile Details:** Location: Z100SC10, Schema: MVNXTEST, Host: s0w1.fritz.box, Port: 5125
- Lines Limit:** 0
- Timeout:** 60
- Locale:** EN
- User Preference:** (empty)

At the bottom of the preferences panel, there are '+ Import' and 'Export' buttons, and a '< Back to Use Cases' button. A red callout bubble is overlaid on the 'Profile' dropdown and the 'Profile Details' section, containing the text: 'This time I'd like to run my analysis against a different system. All Db2 systems defined can easily be chosen from the App's preferences.'

The modern way to visualize BP/GBP problems



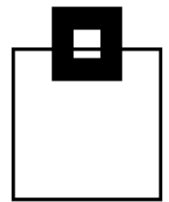
Modern sunburst chart enabling drill through.

The modern way to visualize BP/GBP problems

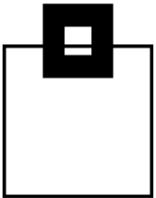
The screenshot displays the 'Bufferpool overview new' interface. At the top, there are tabs for 'Overview', 'KPIs', 'New BP', and 'BP history'. The 'Overview' tab is selected. Below the tabs, the 'Current Status' section shows a traffic light style summary: a green checkmark for '63 OK _ no violations', a yellow warning icon for '4 Warning', and a red 'X' icon for '25 Critical'. Below this, there are tabs for 'BP0', 'BP16K0', 'BP32K', and 'BP8K0', with 'BP0' selected. A list of recommendations is shown below, each with a traffic light icon on the right:

- Change frame size to 1M or 2G to fit the buffer pool size
- Prefetch size (62.50MB): Increase BP size to reach at least 320MB for prefetch unless there is no desire for prefetch in this pool.
- Pagefix Attribute: Alter to PGFIX(YES).
- Calculated VPSEQT Size, the VPSEQT should be altered from 80 to 10.
- Number of Page ins for read

Overview shows a traffic light style of all results for instant problem finding.



The modern way to visualize BP/GBP problems



Bufferpool overview new

- Overview
- KPIs**
- New BP
- BP history

- BP0**
- BP16K0
- BP32K
- BP8K0

Bufferpool BP0 details

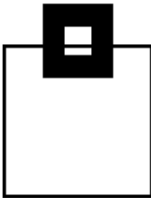
Size :	20000
System Residency Time :	7716
Random Residency Time :	59189
Sequential Residency Time :	6337
Hit Ratio :	95.13
Application :	99.75
System :	90.52

Last Extract : 2024-03-11-08.46.05.003049

Under KPIs you get all the KPIs. If a yellow band is there, it means you have simulation running in this Bufferpool.



The modern way to visualize BP/GBP problems



Calculated VPSEQT Size:	10
Random sync I/Os per second :	0.06
Prefetch size :	62.50
Number of page updates for each page written :	23.45
Number of pages written for each write I/O :	27.04
Page arrival rate (I/Os per sec) :	2.59
I/O intensity :	1574.38
Buffer Reduction :	32
Frame Sizing :	YES
Frame LFAREA :	NO
Pagefix Attribute :	NO
Number of Page ins for read :	30583
Number of page ins for write :	0
Bufferpool paging Page ins > BP size :	YES

More KPIs and now the action buttons appear.

Resize Details Change Object List Simulate VPSEQT



The modern way to visualize BP/GBP problems

BP16K0 Resize

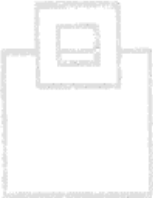
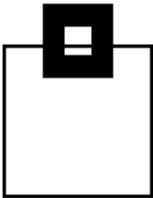
Current Size: **500** 16K Pages

New BufferPool Size :

Here you get the Current Size and the New Size. If you action it, a pop-up appears with the ALTER command.

BP16K0 Resize ×

```
-ALTER BUFFERPOOL(BP16K0) VPSIZE(576)
```



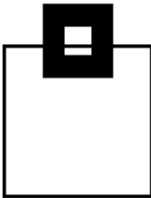
The modern way to visualize BP/GBP problems

BP16K0 Details

SUBSYSTEM_MEMBER	DD10	B401_BUFFERPOOL_ID	120	B401_BUFFERPOOL_NAME	BP16K0	
B402_BUFFERPOOL_SIZE	500	B402_BPOOL_ALLOCATED	500	B404_THRESHOLD_VP_SEQUENTIAL	80	
B404_THRESHOLD_SP_SEQUENTIAL	0	B404_THRS_VRT_DEFERRED_WRT_PCT	5	B404_THRS_VRT_DEFERRED_WRT_ABS	0	
B404_T	B420_ASYNC_WRITE_IO	7349	B420_SYNC_WRITE_IO	1674	B421_DWT_HIT	0
B411_V	B421_DWT_HIT	0	B421_VERTICAL_DWT_HIT	1252	B421_VERTICAL_DWT_HIT	0
B411_S	B546_PREFERRED_FRAME_SIZE_2		B546_USED_FRAME_SIZE_2		B546_BUFFERS_ALLOCATED_2	
B411_S	B546_PREFERRED_FRAME_SIZE_3		B546_USED_FRAME_SIZE_3		B546_BUFFERS_ALLOCATED_3	
B432_F	SYSTEM_HIT_RATIO	69.55	APPLICATION_HIT_RATIO	99.04	SYSTEM_PAGE_RESIDENCY	9350
B432_F	RANDOM_PAGEU_PER_WRITEIO	4	RUN_TIMESTAMP	2024-03-11-08.45.58.050000	SECONDS_ACTIVE	2757597
B546_F	GETPAGE_THRESHOLD_DEFERRED_WRITE	30	THRS_VRT_DEFERRED_WRT_PCT	5	RANDOM_GETPAGE	253877
BUFFER_I	SYNC_READ_IO_R	1853	SEQ_GETPAGE	230422	SYNC_READ_IO_S	2813
S		9658	SEQ_PREFETCH_IO	9147	SEQ_PREFETCH_PAGES_READ	138719
TS		1	LIST_PREFETCH_IO	0	LIST_PREFETCH_PAGES_READ	0
REQUESTS		4573	DYNAMIC_PREFETCH_IO	1200	DYN_PREFETCH_PAGES_READ	4079
IO_BUFFER		0	B415_PREF_DISABLED_NO_READ_ENG			0

Under Details you get all the details!

The modern way to visualize BP/GBP problems



BP16K0 Change



VPSIZE

VPSIZEMIN

VPSIZEMAX

VPSEQT

VPPSEQT

AUTOSIZE

PGSTEAL

FRAMESIZE

DWQT

VDWQT

PGFIX

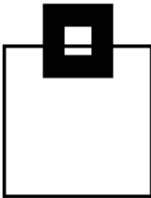
SPSIZE

SPSEQT

Change shows all changeable attributes with built-in logic checks to stop any inadvertent settings.



The modern way to visualize BP/GBP problems



BP16K0 Simulation



Current Size: **500** 16K Pages , Simulated Size : **0**

New Simulated Size :

SPSEQT :

Simulation is like
Resize but just for
simulated buffer
pool pages.

BP16K0 Simulate

```
-ALTER BUFFERPOOL(BP16K0) SPSIZE(100)
```

✓ Confirm



The modern way to visualize BP/GBP problems

BP16K0 VPSEQT ×

Current VPSEQT : 80 %

Calculated VPSEQT : 13 %

New VPSEQT :

✓ Confirm

The calculated VPSEQT can be different from your current VPSEQT.

The modern way to visualize BP/GBP problems

In the next release will be the complete Group Bufferpool support including the recursive math required for the INITSIZE calculation.

Here's how it looks in Excel:

Size in MB	GBP	Read Hit Ratio %	Writes failed due to lack of storage	Reclaims for directory entries	Cross invalidations due to directory reclaims	GBPOOLT * DATA PAGES < Changed pages snapshot	Castout %	Space Map	Data Pages	Sync reads XI miss ratio % JC	Sync reads XI miss ratio % RC
257	GBP0	85	0	0	0	No	No	2598	3262	7,39	92,61
2.049	GBP1	26	0	0	0	No	No	4130564	1747699	7,06	92,94
2.049	GBP2	30	0	0	0	No	No	0	0	5,17	94,83
64	GBP3	7	0	0	0	No	No	9233	0	18,15	81,85
128	GBP32K	94	0	0	0	No	No	72715	143	7,09	92,91
1.025	GBP32K1	30	0	0	0	No	No	27636	0	6,18	93,82
256	GBP32K3	7	0	0	0	No	No	31358	0	8,45	91,55
64	GBP8K0	16	0	0	0	No	No	62	380	86,65	13,35
64	GBP8K1	7	0	0	0	No	No	13526	36204	22,61	77,39
32	GBP8K3	1	0	0	0	No	No	38	0	65,29	34,71
64	GBP16K0	34	0	0	0	No	No	834	47	75,68	24,32
64	GBP16K1	17	0	0	0	No	No	15375	0	52,33	47,67
64	GBP16K3	6	0	152788721	143463562	No	No	13031	0	2,55	97,45
32	GBP16K4						Yes	177			

High Spacemap and Data negotiation counts!

Nasty amount of Reclaims and XI's here!

Bad spread of Sync reads XI miss ratios!

The modern way to visualize BP/GBP problems

In the next release will be the complete Group Bufferpool support including the recursive math required for the INITSIZE calculation.

Here's how it looks in Excel:

Total No. Directory entries	Total No. Data entries	Size in MB for data entries	Size in MB for directory entries	Starting size of GBP in MB	Current Size of GBP	Increase GBP by xxx MB	Buffer Pool
250.001	50.001	196	103	299	257	42	BP0
2.222.223	222.223	869	912	1.781	2.049	0	BP1
2.222.223	222.223	869	912	1.781	2.049	0	BP2
50.001	10.001	40	21	61	64	0	BP3
42.858	2.858	90	18	108	128	0	BP32K
177.779	17.778	556	73	629			
44.445	4.445	139	19	158			
44.445	4.445	35	19	54			
44.445	4.445	35	19	54	64	0	BP8K1
41.668	1.667	14	18	32	32	0	BP8K3
42.858	2.858	45	18	63	64	0	BP16K0
42.858	2.858	45	18	63	64	0	BP16K1
44.445	4.445	70	19	89	64	25	BP16K3
40.743	741	12	17	29	32	0	BP16K4

Two bad INITSIZE problems found!

The modern way to visualize BP/GBP problems

With one click you generate all of the ALTERs you require and, naturally, we do not execute them!

You can “batch up” the changes and issue them at a quiet time.

Please remember:

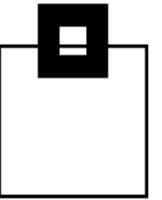
Measure your system first.

Do the ALTERs – Perhaps not **all** of them at once!

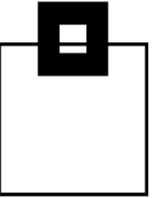
Measure your system.

You ***should*** see a system-wide improvement and/or application-level improvement.

This is all done ****without**** even knowing the applications that are running!



The modern way to visualize BP/GBP problems



Recommended further reading:

[Db2 12 Tuning database buffer pools](#) IBM Db2 Documentation

[Db2 11 for z/OS Buffer Pool Monitoring and Tuning](#) A great Redbook

[DB2 9 for z/OS: Buffer Pool Monitoring and Tuning](#) Another great Redbook

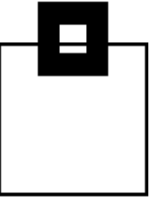
[Db2 for z/OS Hot topics and Best practices](#) John Campbell

[Db2 12 for z/OS Buffer Pools](#) Robert Catterall

[Best Practices for DB2 on z/OS Performance](#) Dan Luksetich and Susan Lawson



Agenda



- What use are BUFFERPOOLS?
- Is it worth tuning?
- How do you tune them?
- What about GROUP BUFFERPOOLS?
- The modern way to visualize BP/GBP problems
- **Q&A**



Questions & Answers

