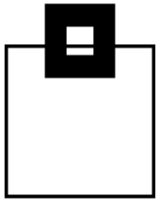
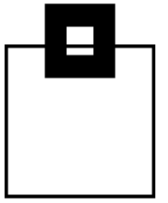


BUFFERPOOL Tuning
The Next Generation

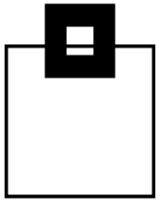
Roy Boxwell,
SEGUS Inc.

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

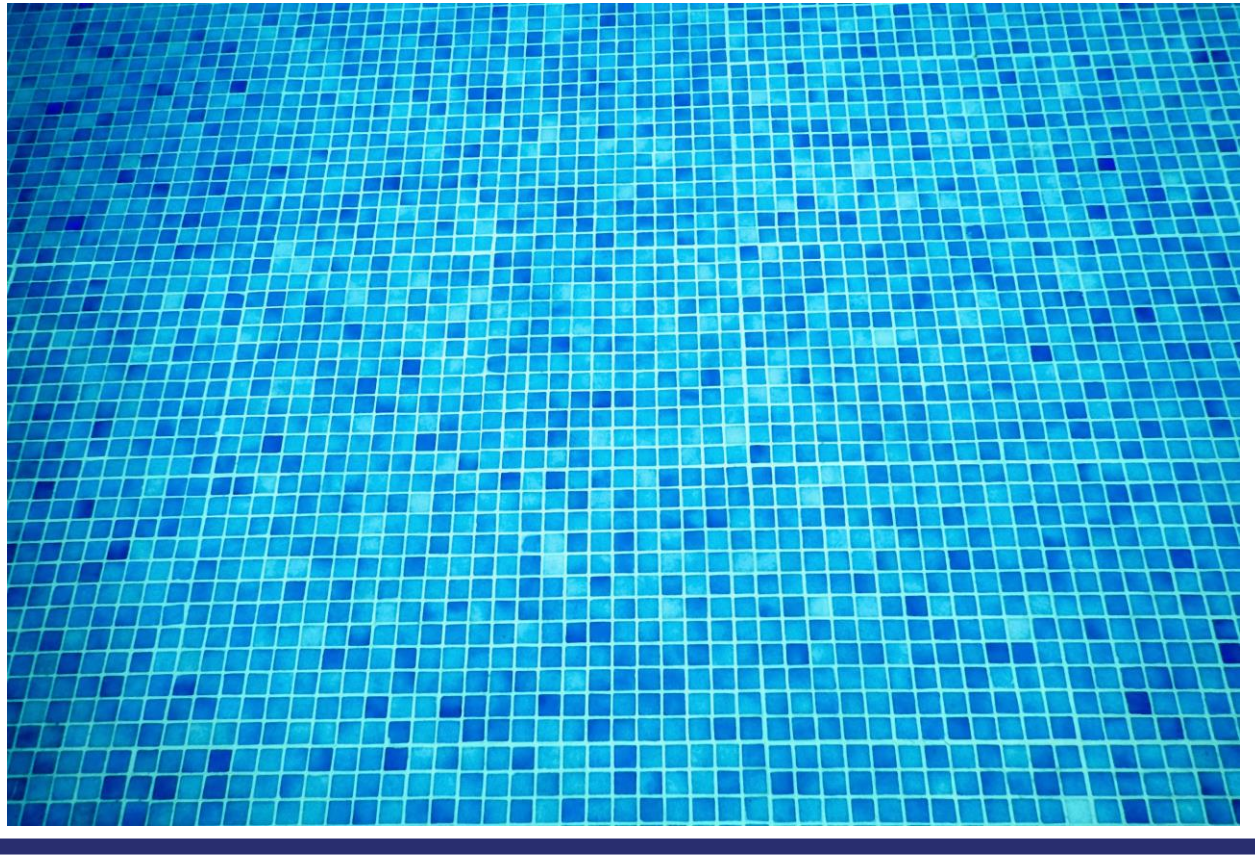
Agenda



- 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!)

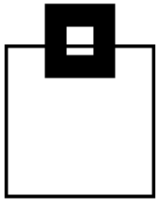
The idea, back in the day, was to have two sizes of BP for matching the two sizes of DB2 pages - namely 4KB and 32KB.

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
			_____	1280K	320K

Figure 29. Buffer Pool Size Calculation

What use are BUFFERPOOLs?



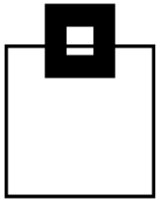
In DB2 V3.1 we got the boost of BP0 to BP49 and BP32K, BP32K1 to BP32K9. BP0 had 2,000 pages as a default and BP32K only 24(!) pages as a default…

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
```

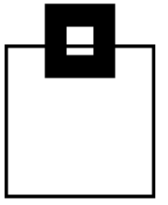
What use are BUFFERPOOLs?

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 poo
    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 cont
```

```
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?

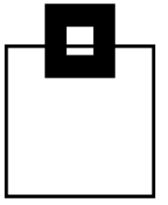


Ok, so we have a nice set of default sizes but what are they used for?

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?



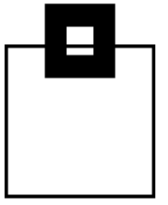
Why do we have so many BPs in Db2 for z/OS? Why not just one huge area of RAM stuffed full of data?

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?



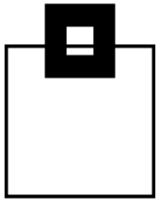
Why do we have so many BPs in Db2 for z/OS? Why not just one huge area of RAM stuffed full of data?

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?

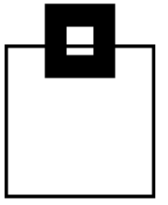


Now imagine an application process that is randomly reading data through an index. It fetches the leaf and non-leaf pages into the BP as it needs them and then carries on.

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?

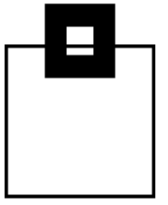


Now imagine an application process that is randomly reading data through an index. It fetches the leaf and non-leaf pages into the BP as it needs them and then carries on.

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!

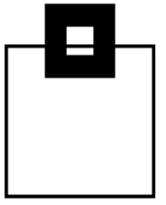
What use are BUFFERPOOLs?

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?

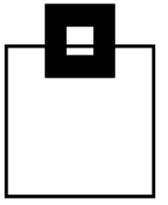


Now imagine a sort – Yes I am talking about DSNDB07 usage here! You might not know it but SORT requires a BP as well. What are the odds of a repeated reread in a sort pool?

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?

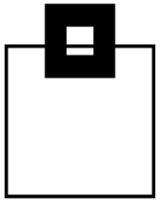
"It depends" raises its ugly head here!

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?

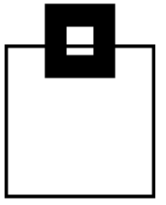


Most shops have no personnel to look at, monitor, tune and change the BPs at their site. The expertise is “graying” and lots of people are “afraid” of changing something so crucial as a BP.

This is not healthy!

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

What use are BUFFERPOOLs?

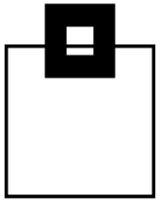


The Db2 Directory and Catalog are the most important part of any Db2 system. They are the meta-data repositories of the entire system and contain *everything* you need to run, but these objects go into only the listed default BPs and I will bet that most shops also use these BPs for application data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

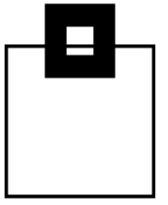
What use are BUFFERPOOLs?



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

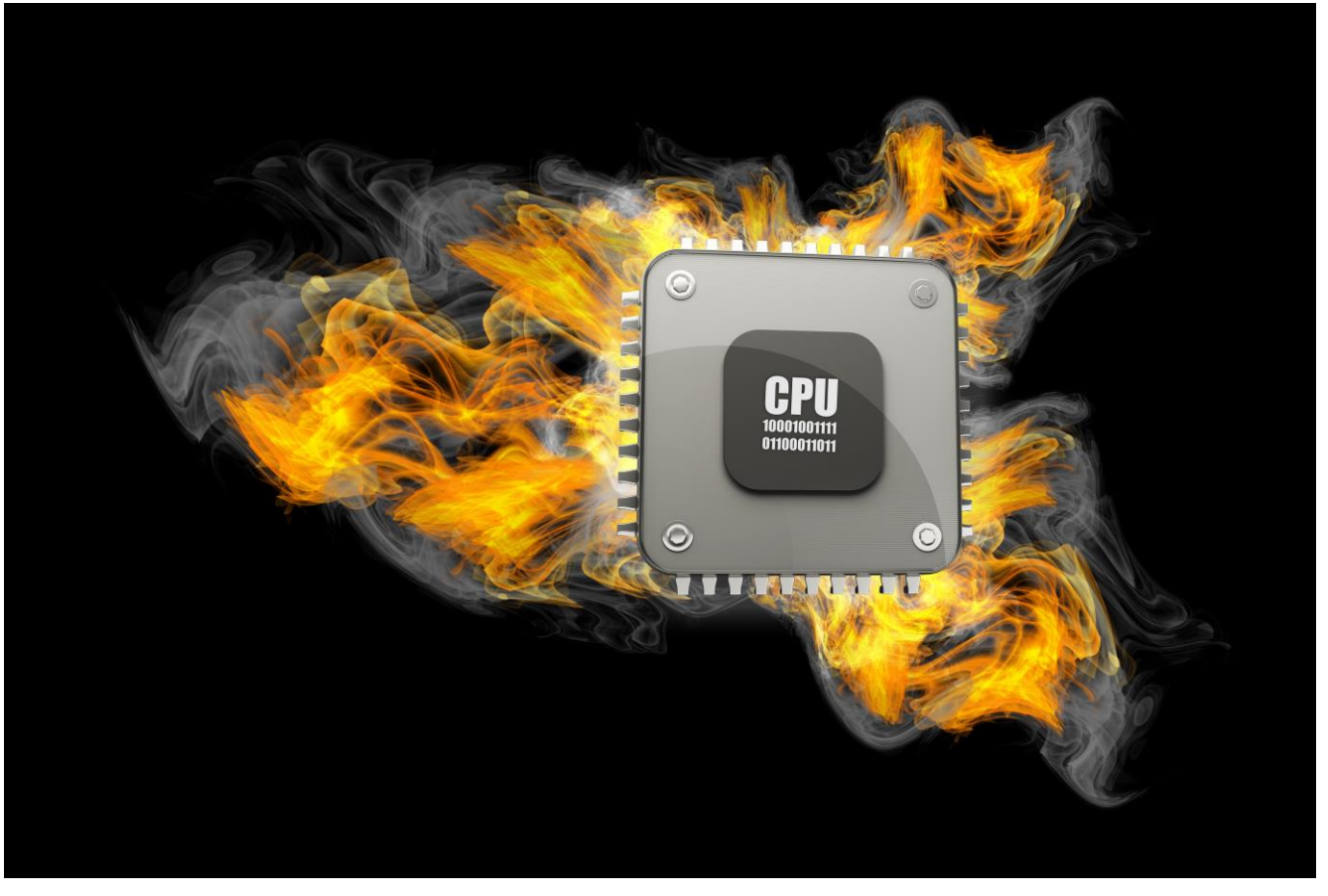
Is it worth tuning?

The answer is a massive

YES!

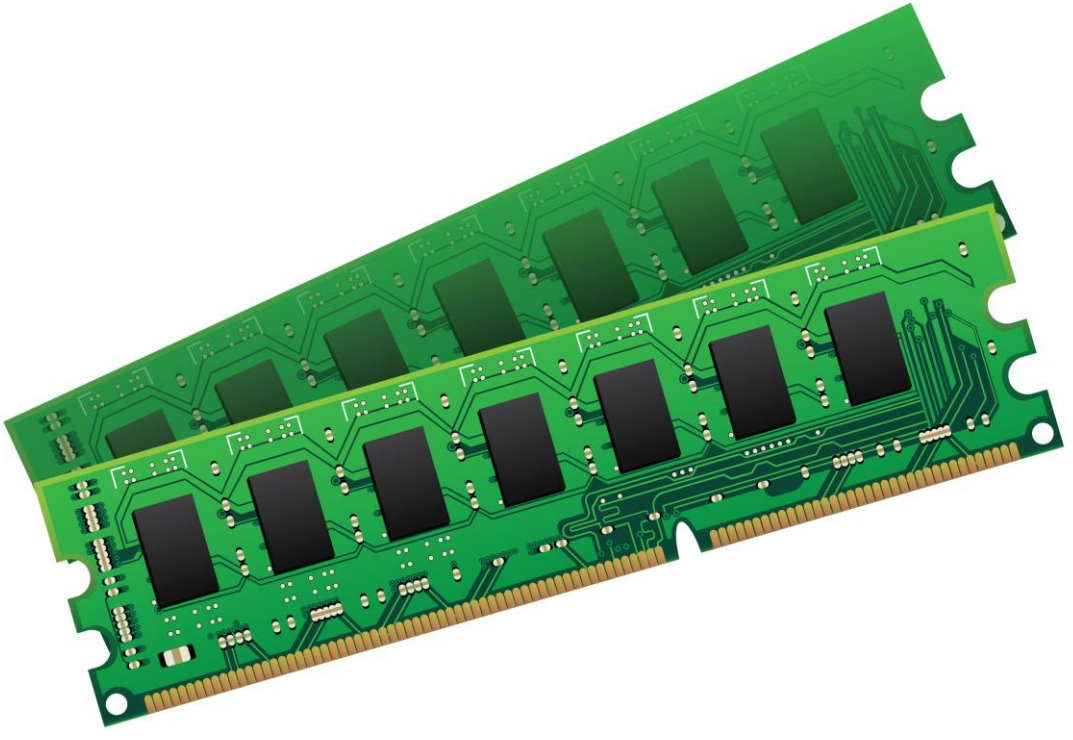
Is it worth tuning?

Your CPU is on fire!



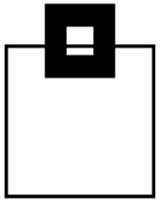
Is it worth tuning?

Your RAM is choked?



www.123rf.com

Is it worth tuning?



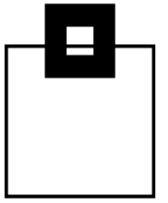
It is often written that the quickest and best results of any type of tuning are indeed with BP and GBPs.

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%.

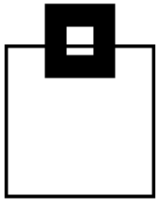
Is it worth tuning?



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?



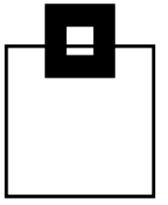
Another very popular problem is the:

"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:

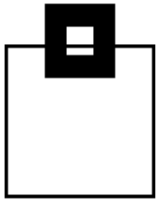
"Stuff it in BP8K0, I know that BP exists!"

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...

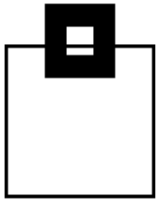
Death by sync I/O is not a pretty sight...

Agenda



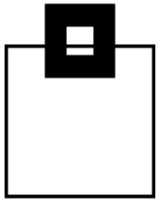
- What use are BUFFERPOOLs?
- Is it worth tuning?
- How do you tune them?
- What about GROUP BUFFERPOOLS?
- The modern way to visualize BP/GBP problems
- Q&A

How do you tune them?



There is an ALTER command...

How do you tune them?

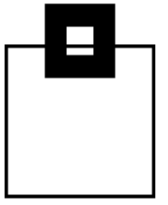


There is an ALTER command...

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?



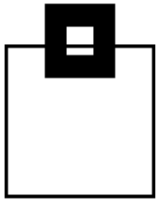
There is an ALTER command…

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...

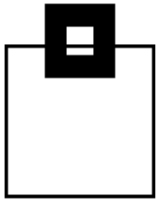
Now in Db2 for LUW you would just throw a few more CPUs or GBs of RAM at the problem and walk away a happy bunny.

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.

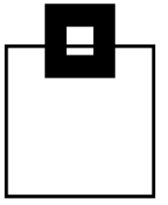
You must then ALTER them to do what they should be doing!

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?

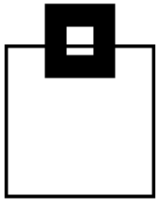


These are the forgotten “zombies” of the Db2 for z/OS BP world!

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?



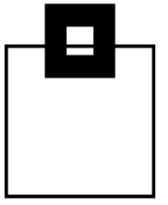
These are the forgotten “zombies” of the Db2 for z/OS BP world!

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!

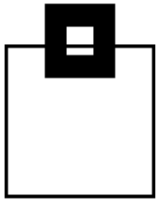
They are incredibly important for the well-running of any data-sharing system and basically require the same style of tuning as normal – local – BPs.

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=xxxxxxxx.yyyyy,KEEP=OLD

Then issue the:

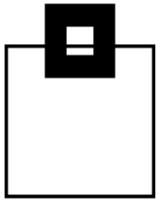
SETXCF START,REBUILD

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

SETXCF START,RB,DUPLEX,STRNAME=xxxxxxxx.yyyyy

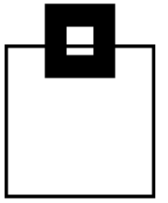
Where xxxxxxxx is Group Name and yyyyy is the Group BP Name.

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

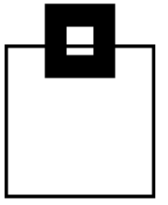


We now know that we have a problem!

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



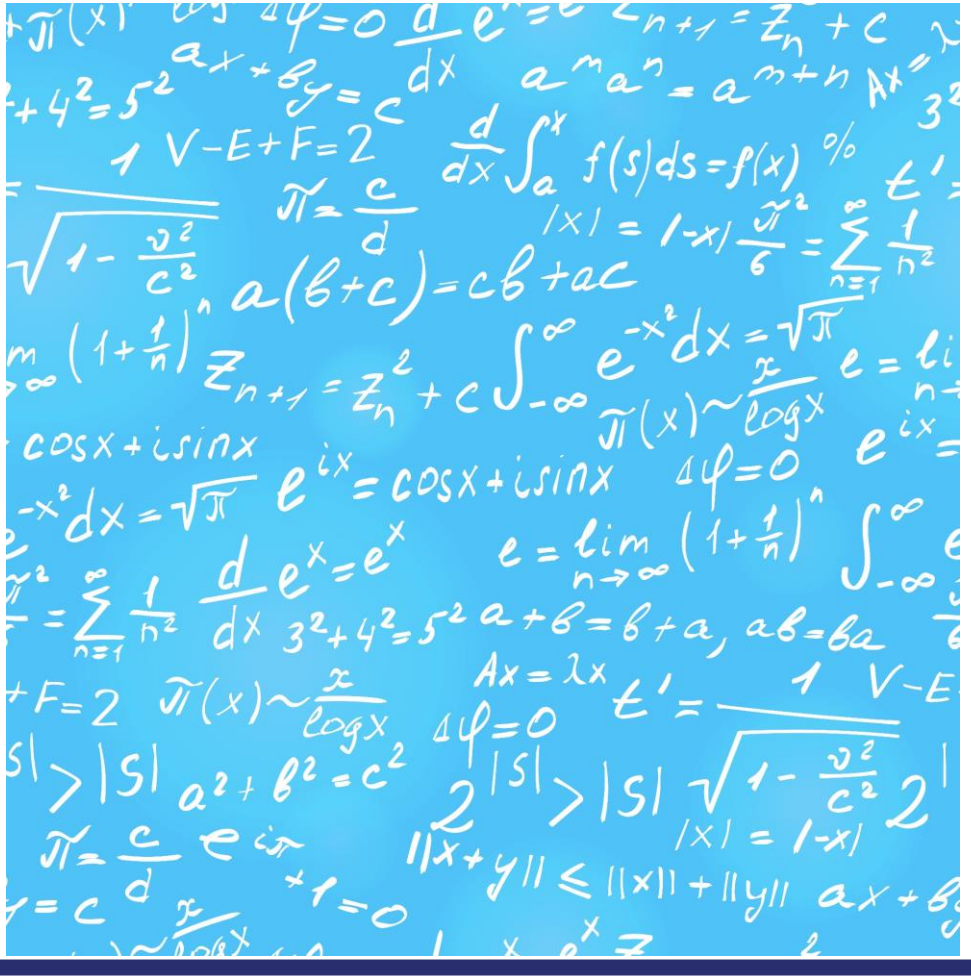
We now know that we have a problem!

How can we actually visualize this and do stuff?

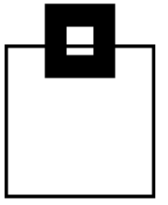
The archaic way that 3270 green screen outputs data and/or the extremely detailed obscure formulae that must be used and/or the different sources of data that must be trawled, all make it “non-trivial”…

How “non-trivial”???

The modern way to visualize BP/GBP problems



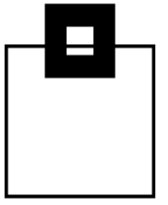
The modern way to visualize BP/GBP problems



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

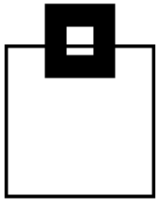
The modern way to visualize BP/GBP problems



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

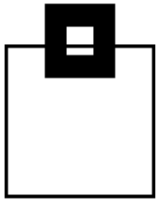
The modern way to visualize BP/GBP problems



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

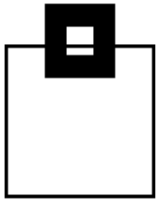
The modern way to visualize BP/GBP problems



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…

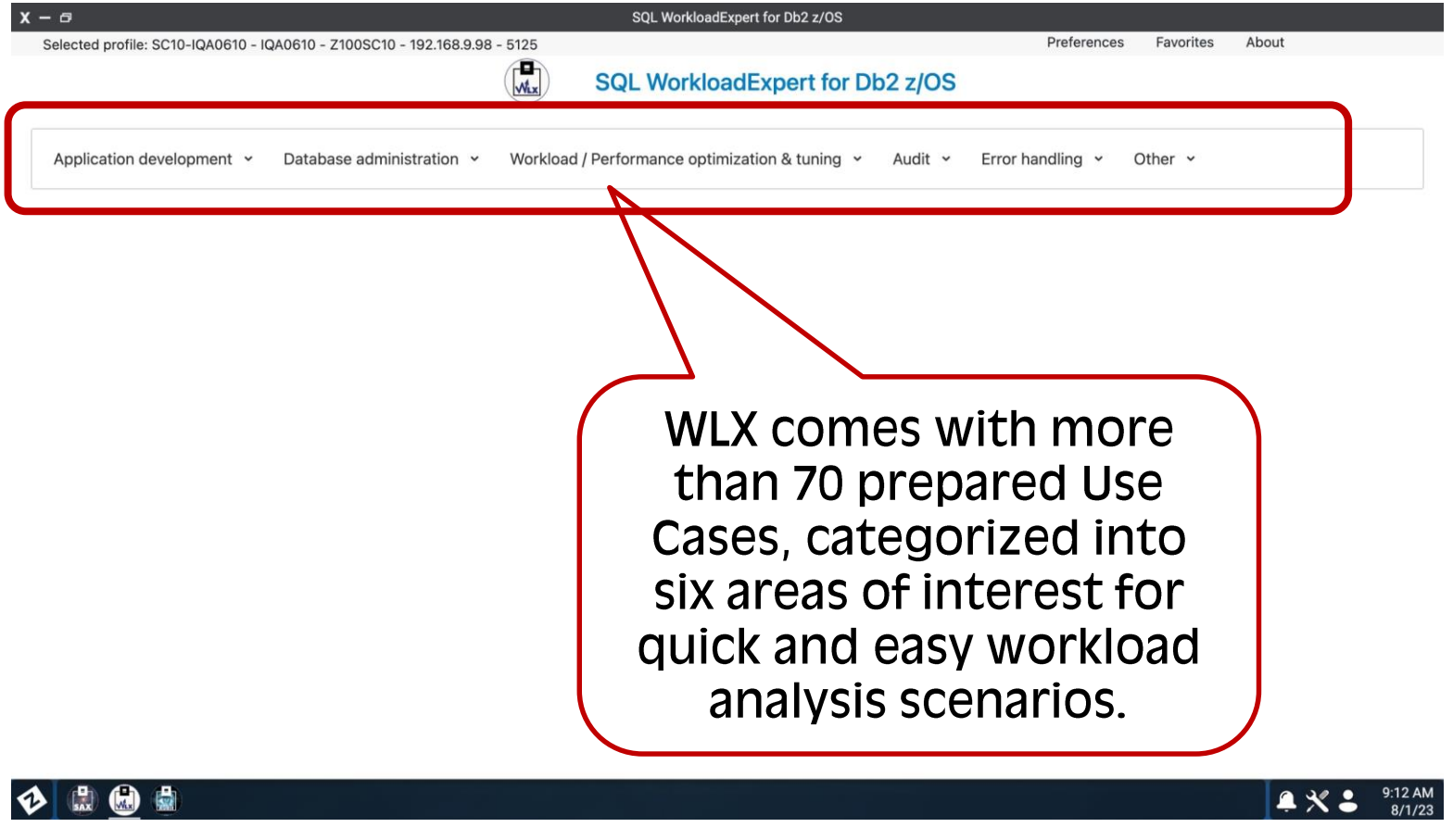
The modern way to visualize BP/GBP problems

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!

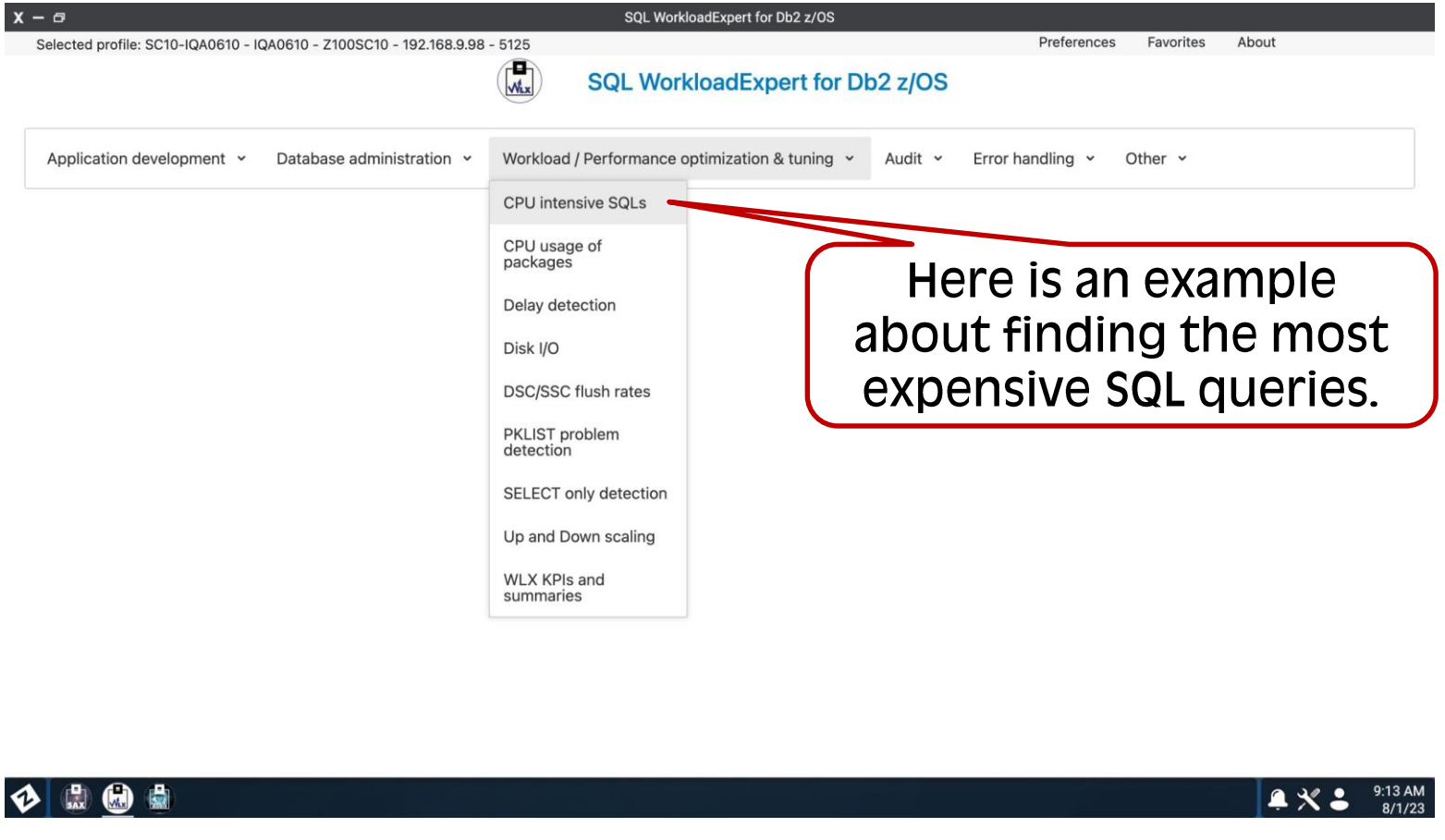
This is all pretty nasty work that someone probably did way back in the 1990's but since then it has not been updated…

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

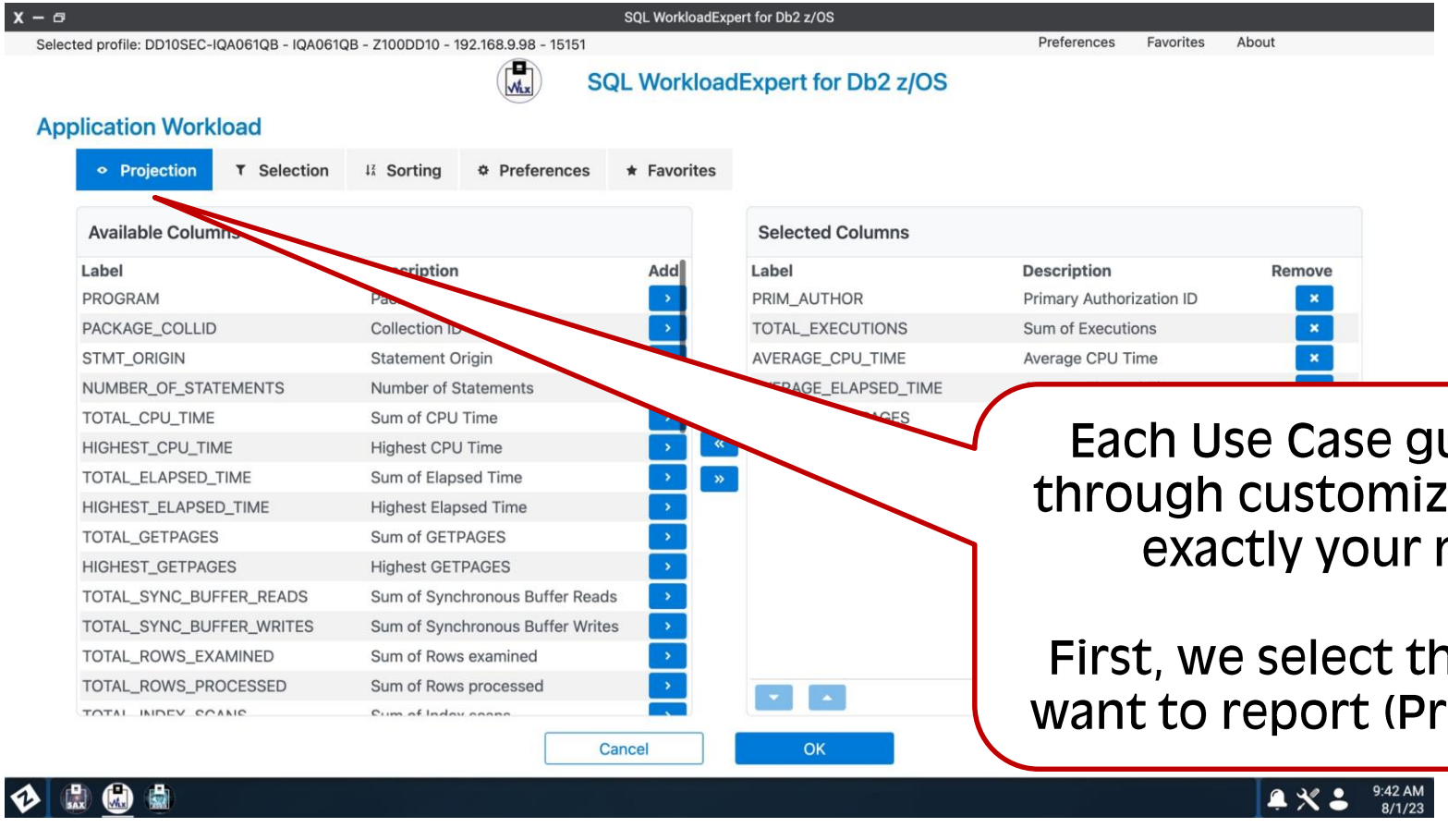
The modern way to visualize BP/GBP problems



The modern way to visualize BP/GBP problems



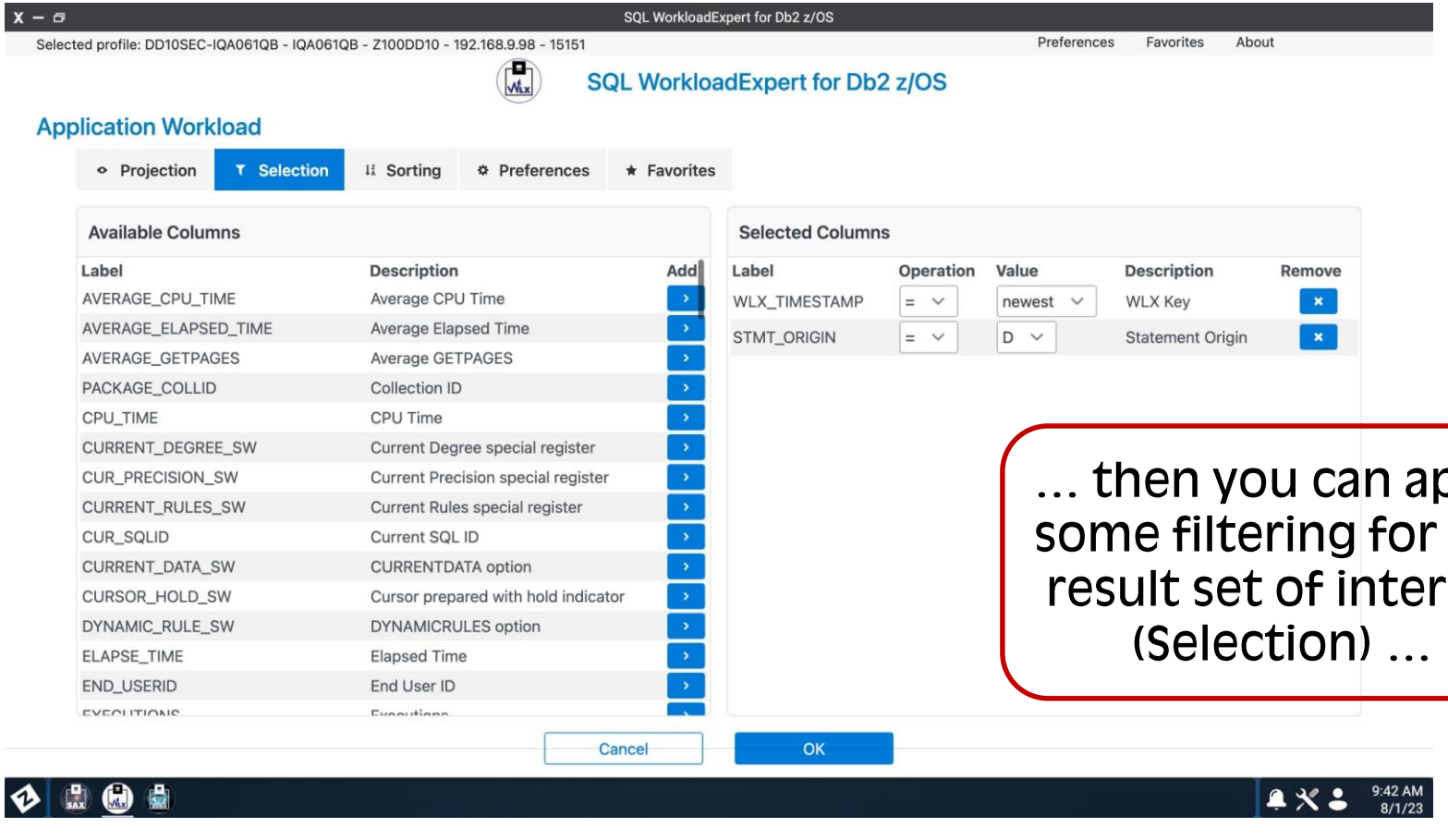
The modern way to visualize BP/GBP problems



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

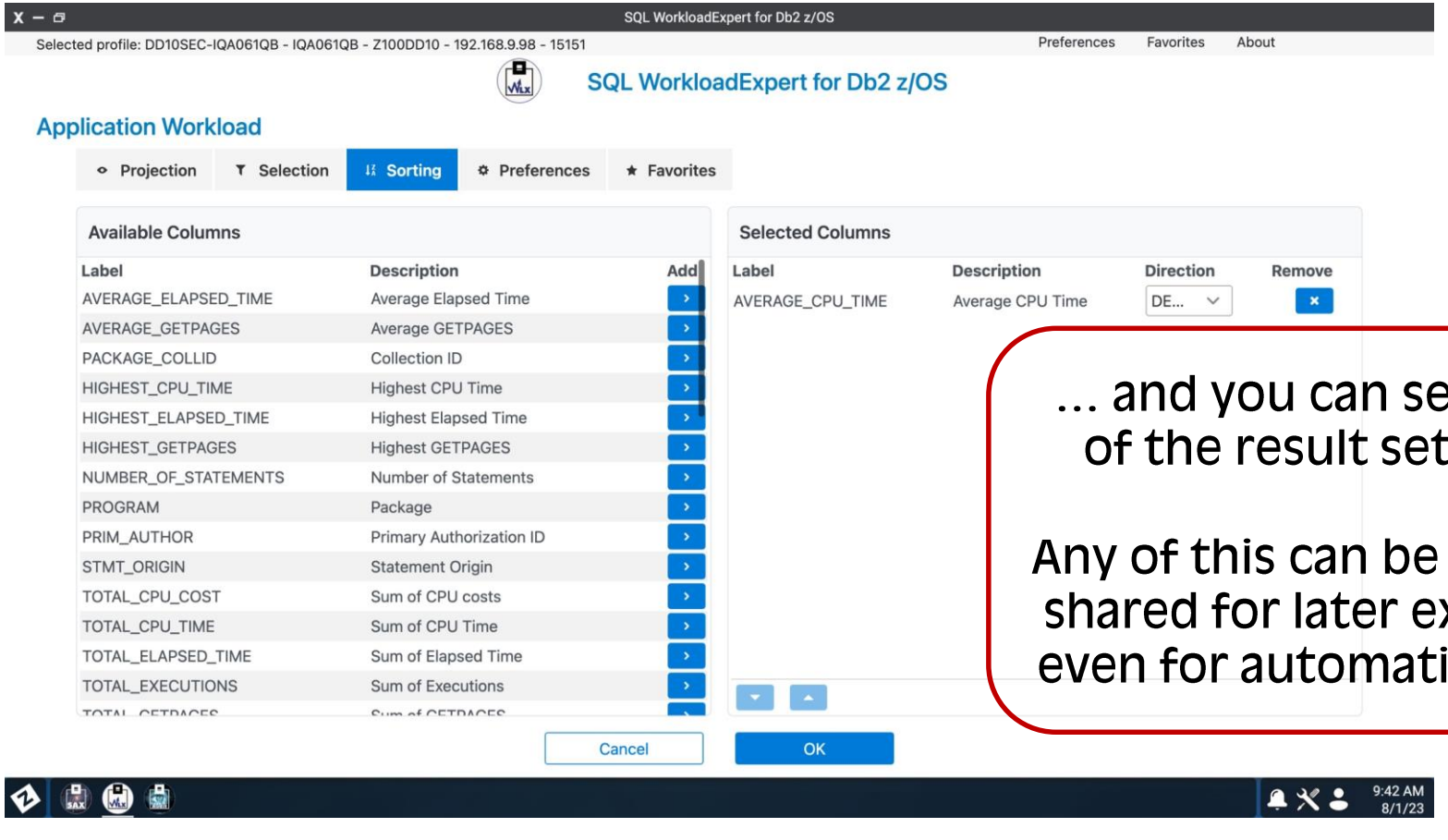
First, we select the data we want to report (Projection)…

The modern way to visualize BP/GBP problems



… then you can apply some filtering for the result set of interest (Selection) …

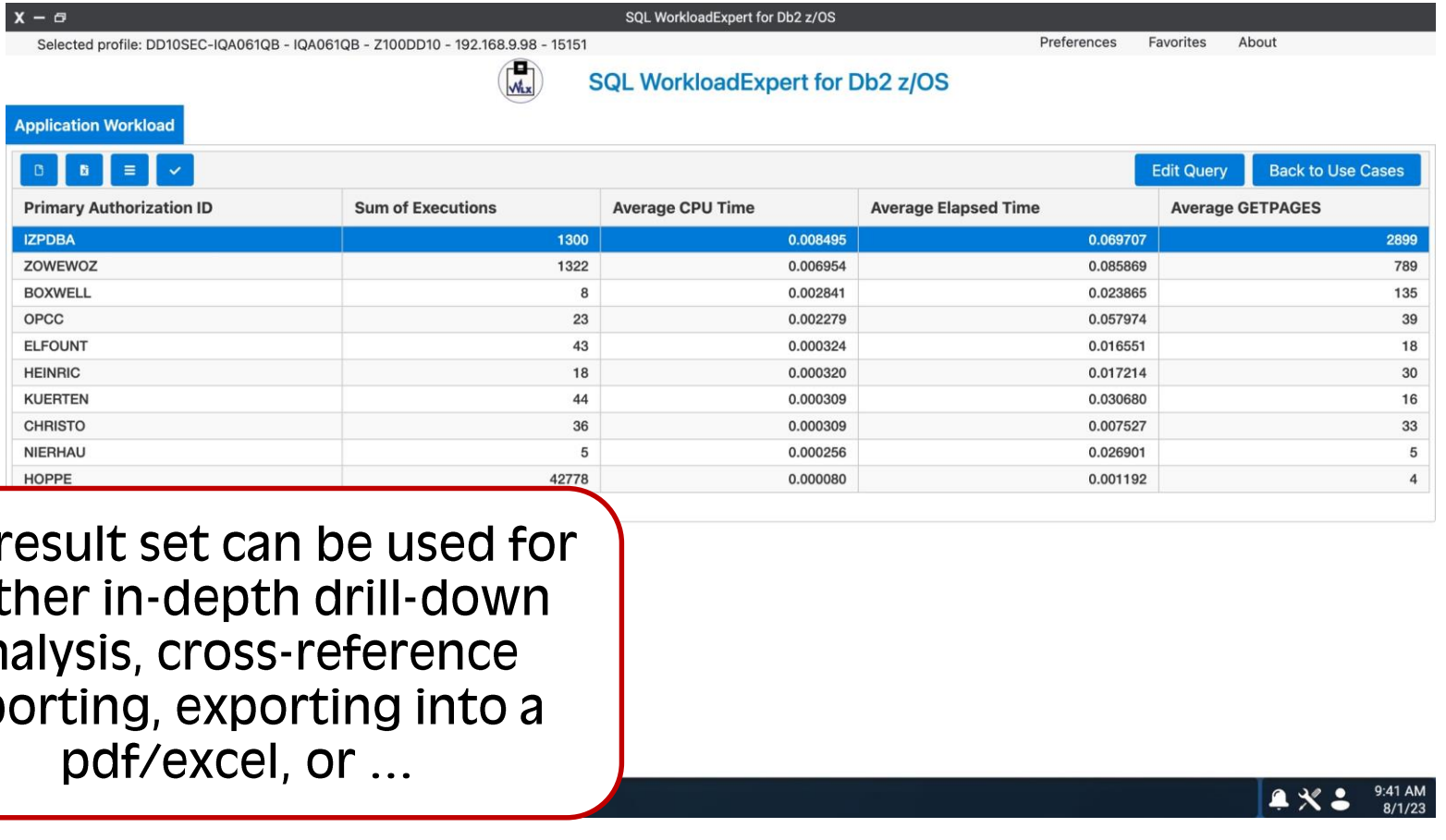
The modern way to visualize BP/GBP problems



... and you can set the order of the result set (Sorting).

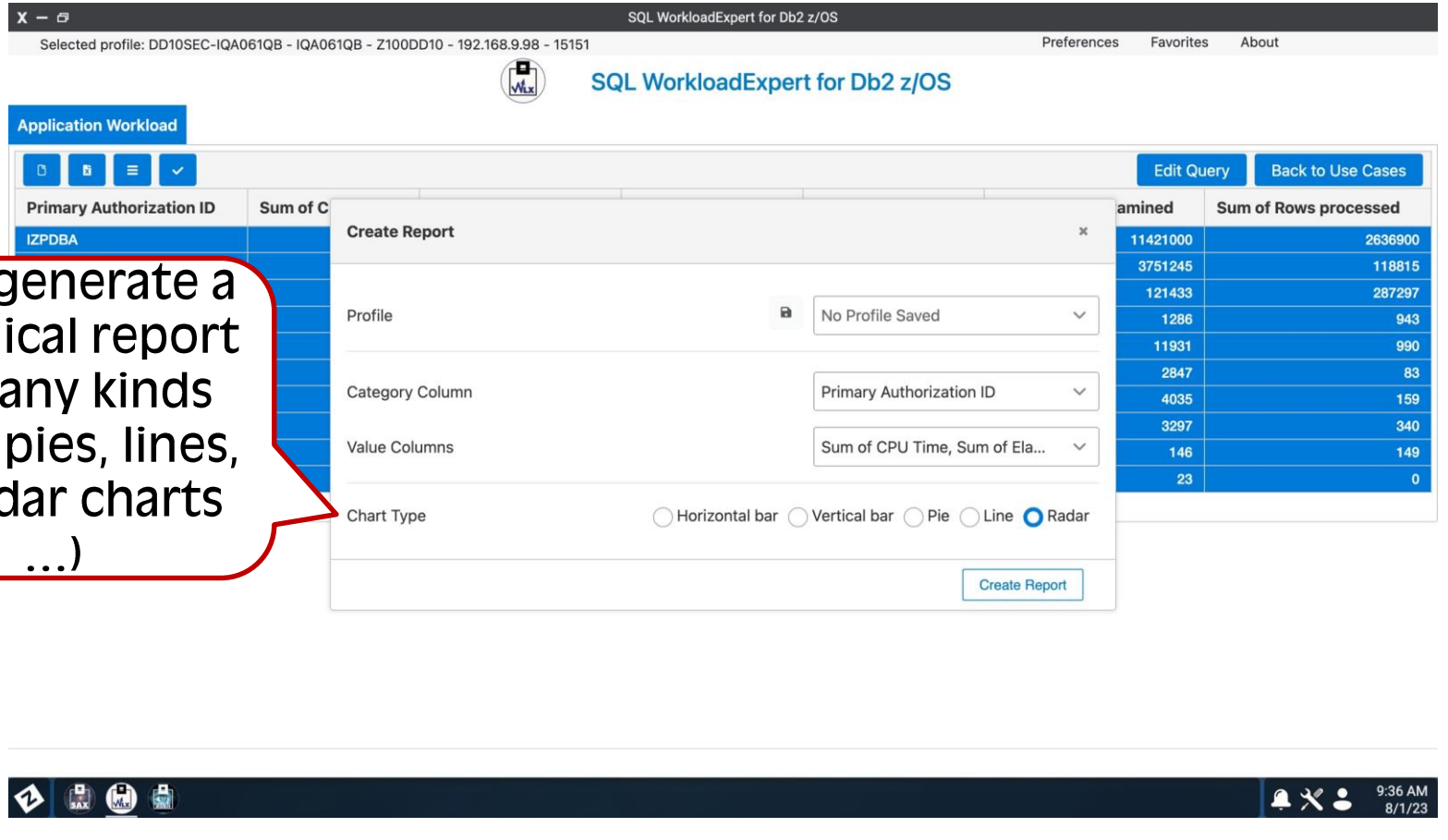
Any of this can be kept and/or shared for later execution, or even for automatic reporting.

The modern way to visualize BP/GBP problems



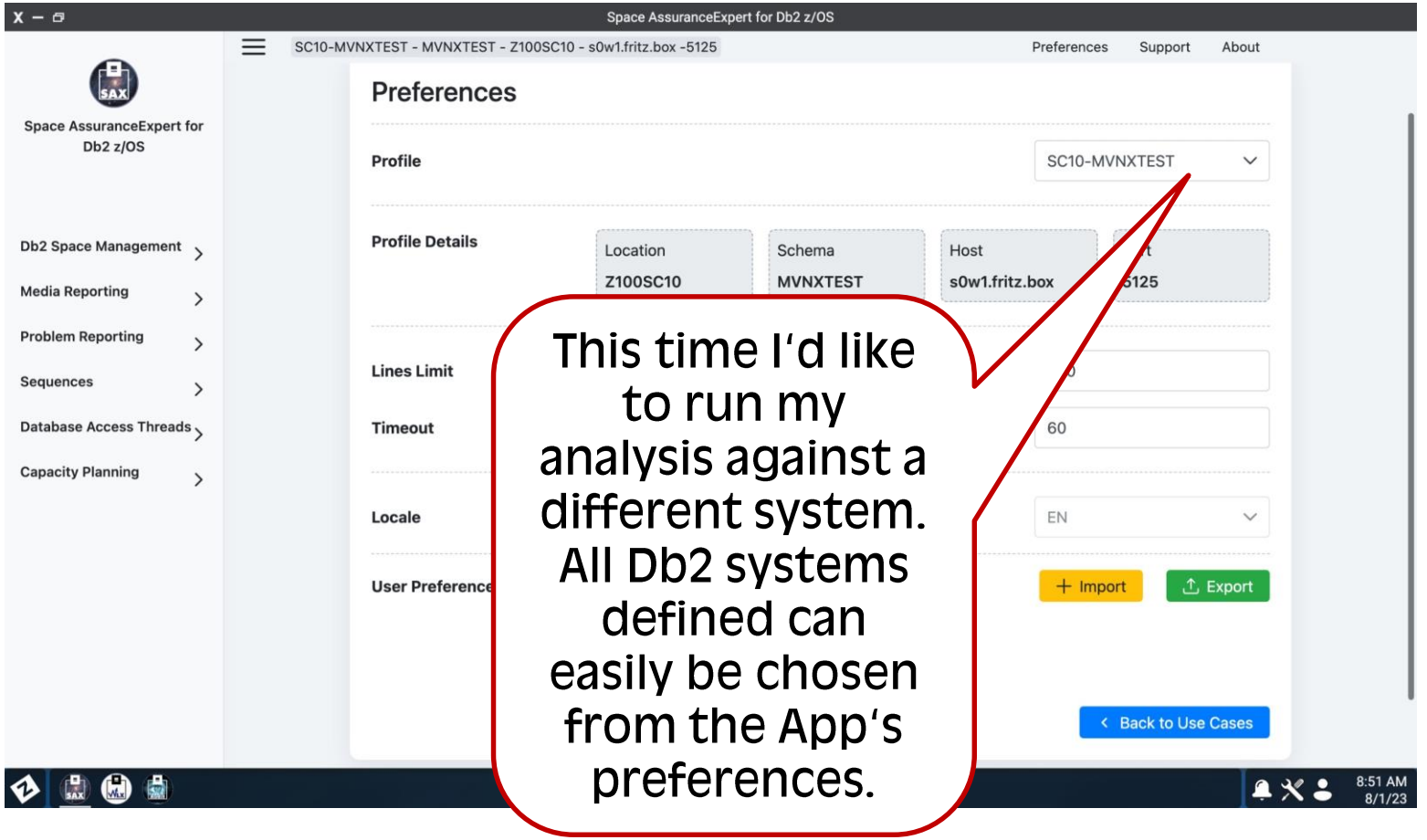
The result set can be used for further in-depth drill-down analysis, cross-reference reporting, exporting into a pdf/excel, or ...

The modern way to visualize BP/GBP problems

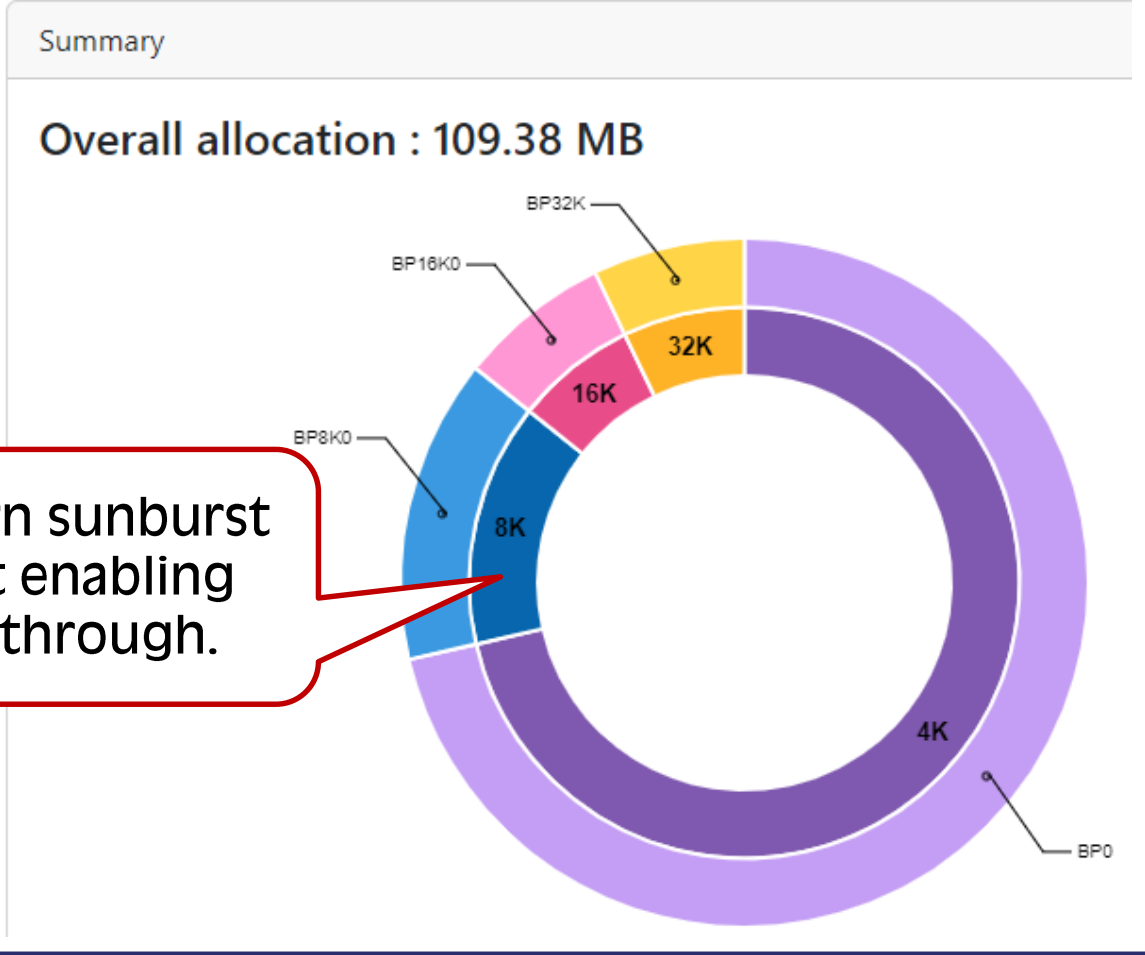


... to generate a graphical report of many kinds (bars, pies, lines, or radar charts ...)

The modern way to visualize BP/GBP problems



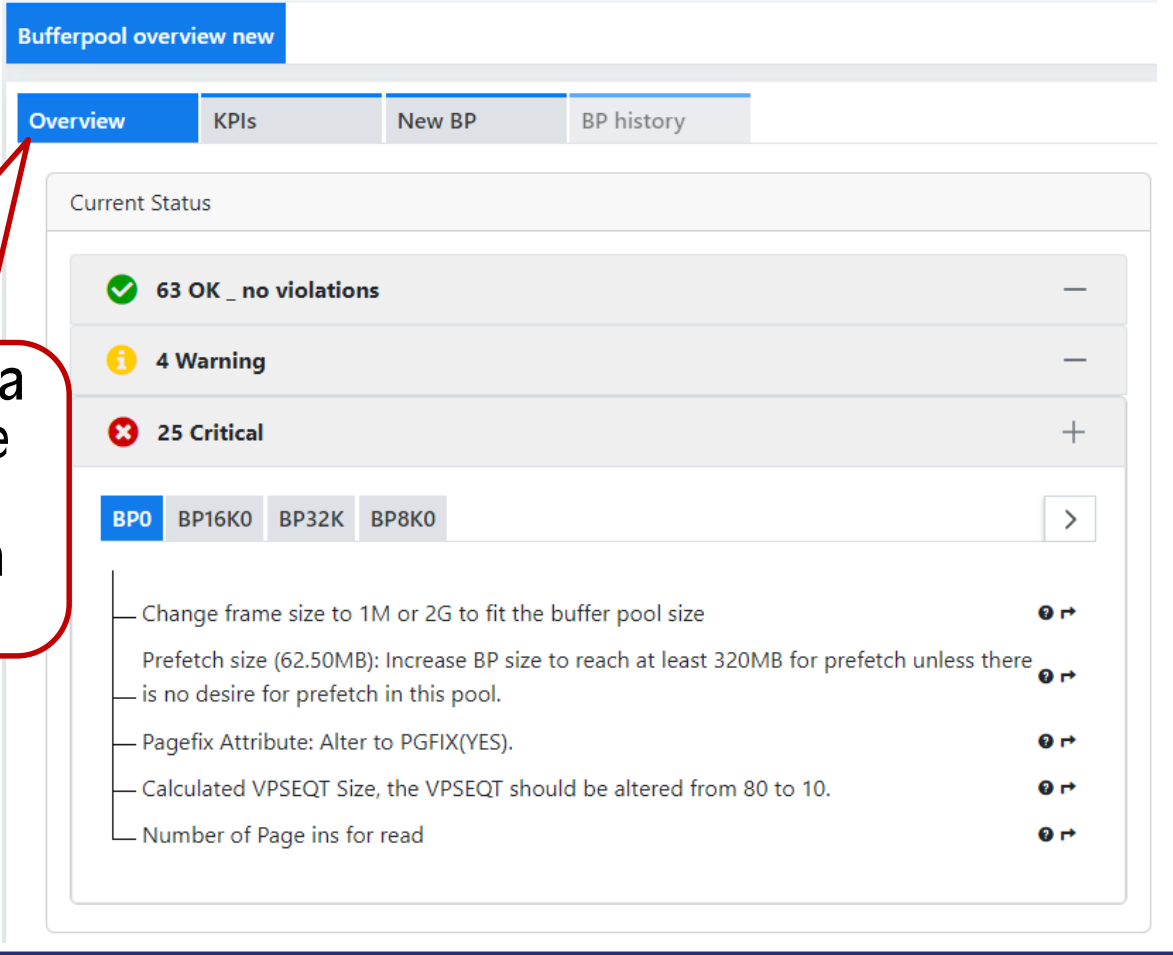
The modern way to visualize BP/GBP problems



Modern sunburst chart enabling drill through.

The modern way to visualize BP/GBP problems

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



The modern way to visualize BP/GBP problems

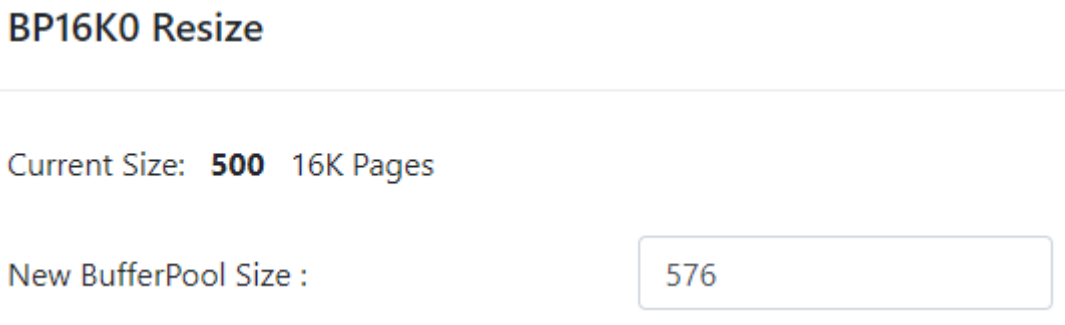
The modern way to visualize BP/GBP problems

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

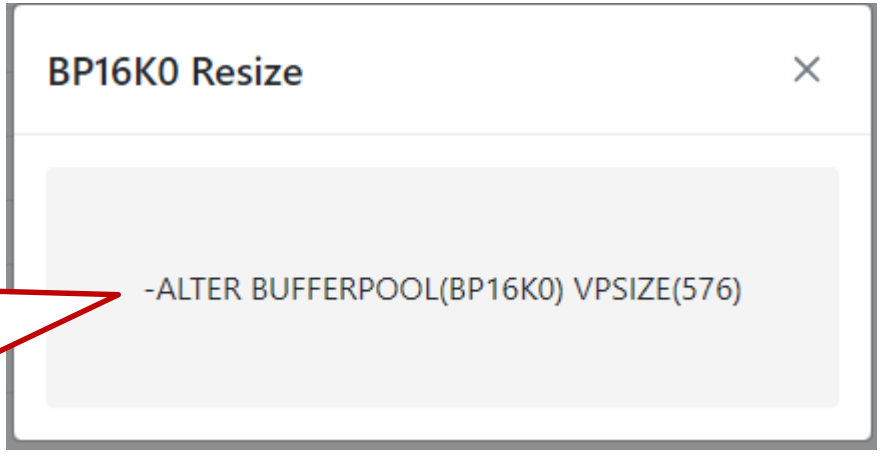
Resize | Details | Change | Object List | Simulate | VPSEQT

More KPIs and now the action buttons appear.

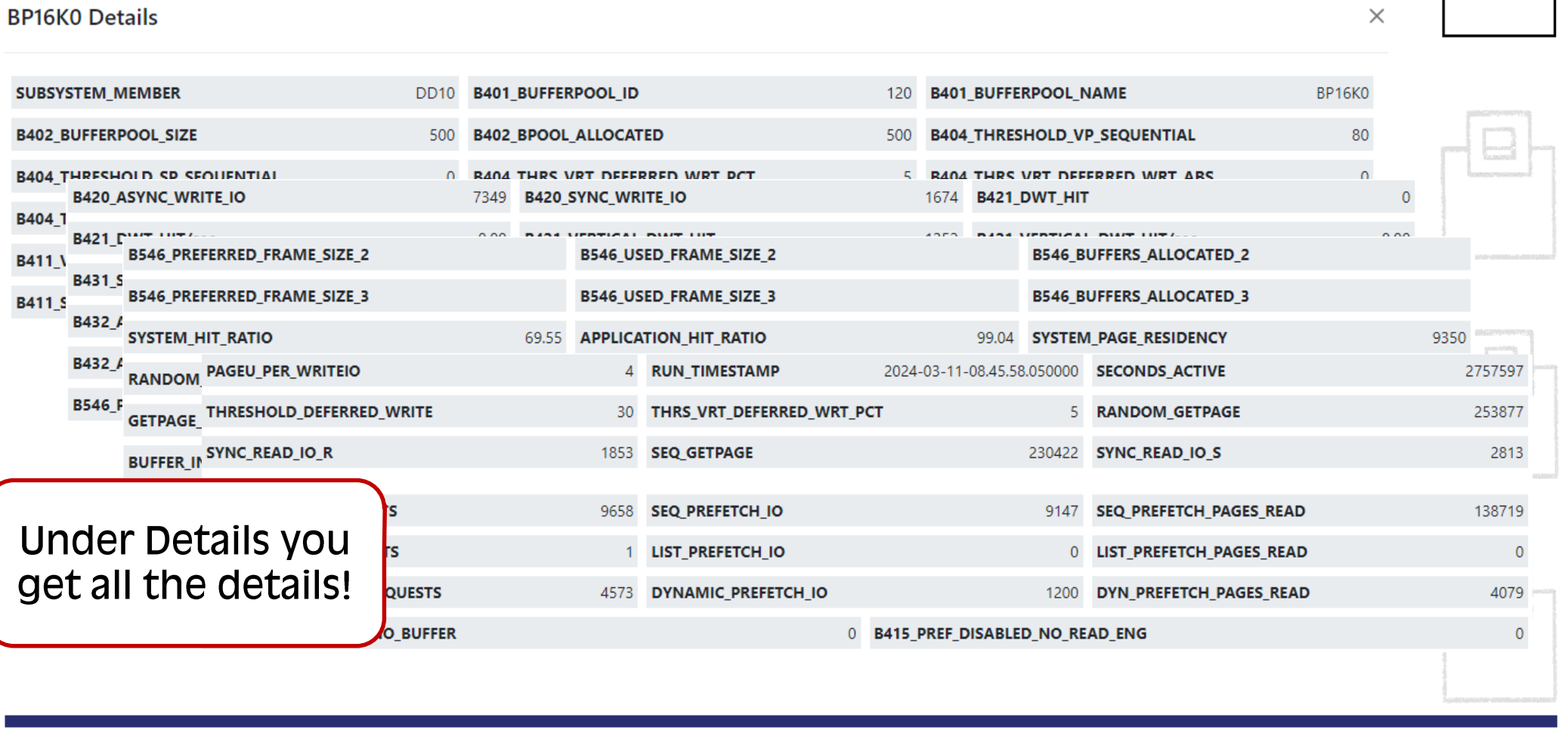
The modern way to visualize BP/GBP problems



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

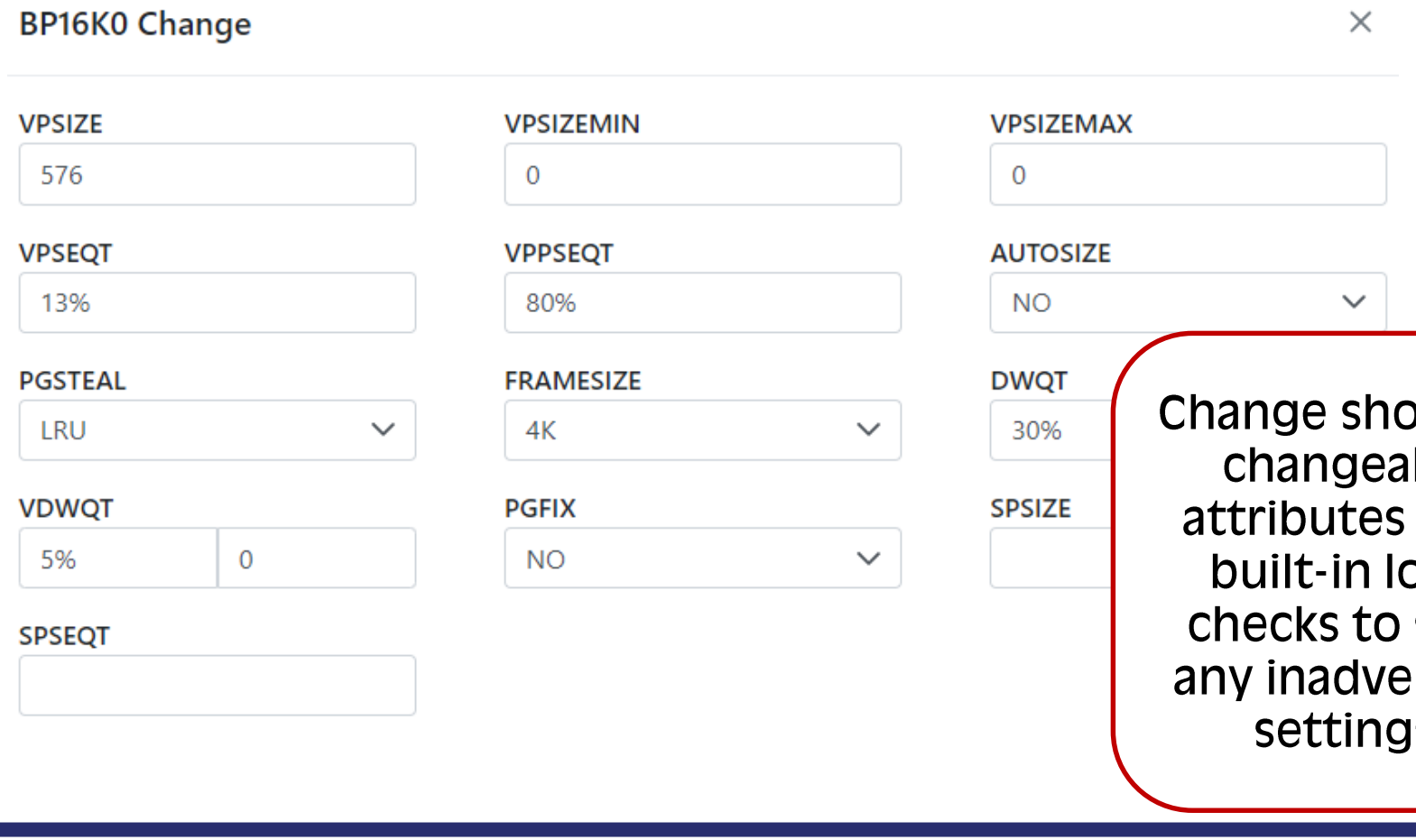


The modern way to visualize BP/GBP problems



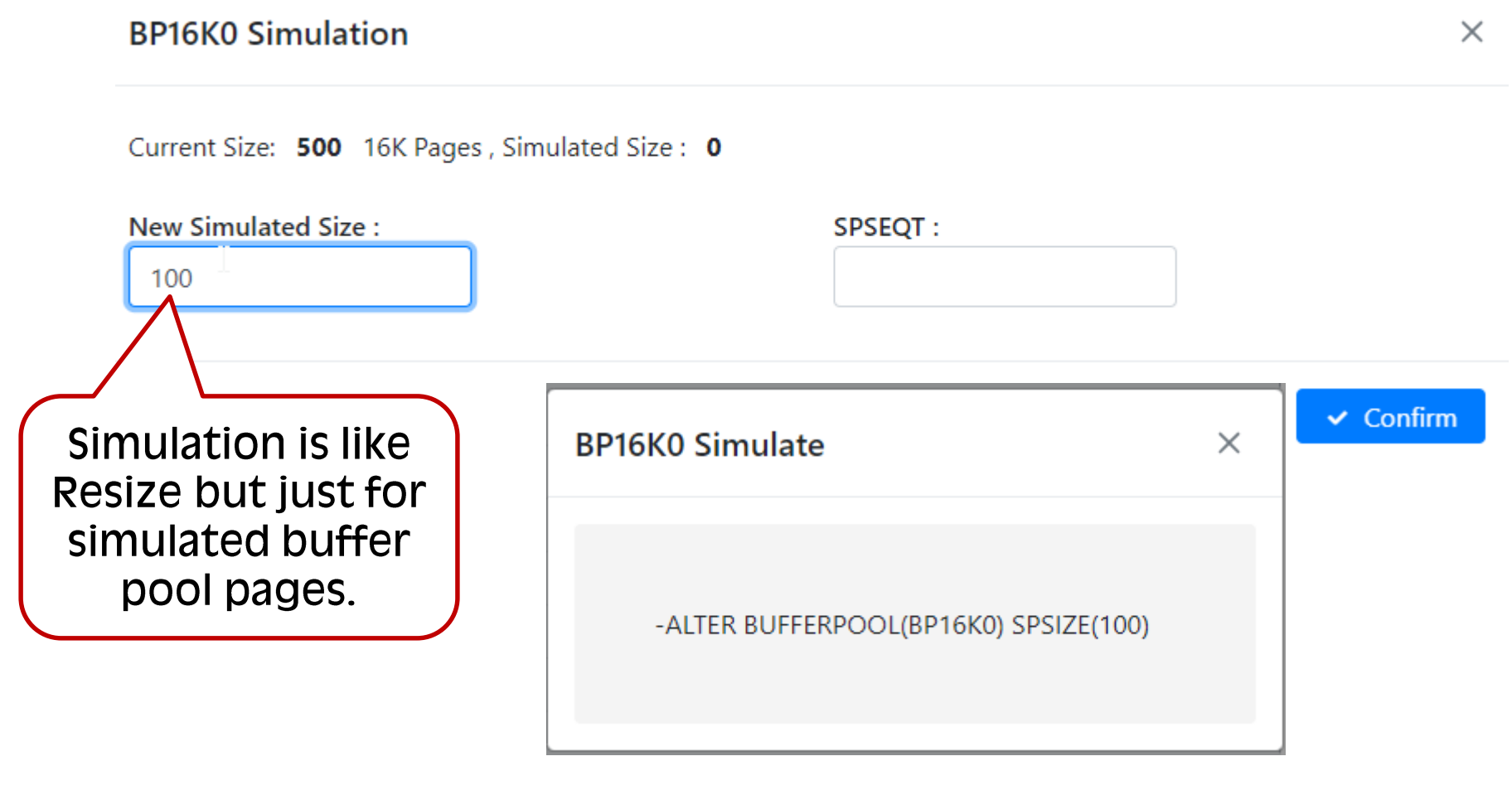
Under Details you get all the details!

The modern way to visualize BP/GBP problems

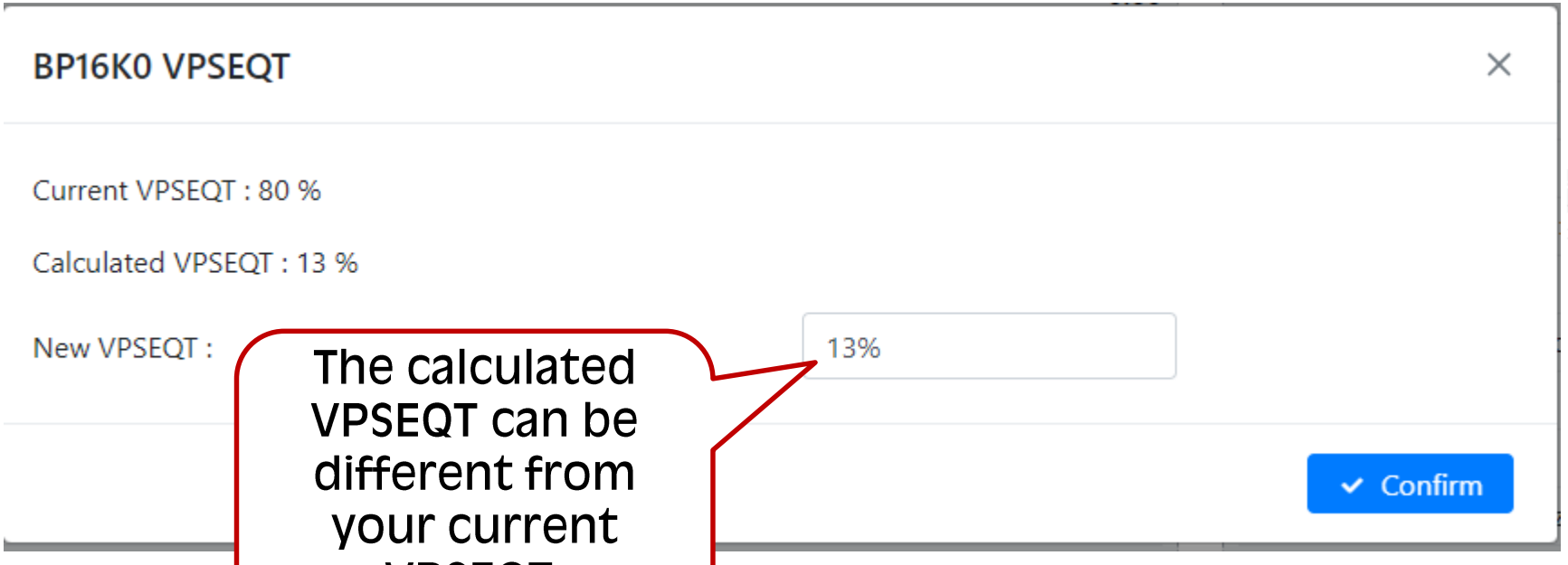


Change shows all changeable attributes with built-in logic checks to stop any inadvertent settings.

The modern way to visualize BP/GBP problems



The modern way to visualize BP/GBP problems



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:

High Spacemap and Data negotiation counts!

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			

Nasty amount of Reclaims and XI's here!

Bad spread of Sync reads XI miss ratios!

The modern way to visualize BP/GBP problems

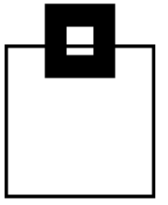
In the next release will be the complete Group Bufferpool support including the recursive math required for the INITSIZE calculation.

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.

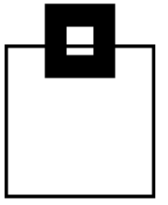
Please remember:

- 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

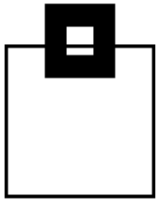


Recommended further reading:

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