

## UNDERSTANDING VDWQT & DWQT PARMS IN DB2.

The parameters governing the Database Asynchronous write warrant special care and proper setting in order to utilize the system resources properly. The concept of Deferred write is as old as the concept of Multitasking. By deferring the writes and letting the OS / DBMS to schedule these at periodic intervals, it is possible to effectively utilize the I/O sub-system dynamics. Apart from I/O utilization, the CPU is forced to do additional work periodically improving the bits per instruction throughput. About 160 I/Os will result in a MIP (approximately). Who wants to waste the MIPS? A brief description of how these parameters can be tuned to achieve an overall improvement in writes throughput is presented here.

VDWQT or the Vertical Deferred write threshold is triggered when a preset percentage of pages pertaining to a single dataset is updated and hence 'IN USE'. Thus any particular dataset is being prevented from monopolizing the Virtual Pool. If the percentage of updated pages for a single dataset exceeds this preset value, writes are scheduled for that dataset, upto 128 pages. This threshold is checked after each update to a page. The default value for this 10% and it can be altered from 0 to 90%.

DWQT is sometimes called the horizontal threshold, applies to the entire Virtual Pool. It is also a preset value expressed in percentage of Bpool, which might be occupied by unavailable pages. The default for DWQT is 50%. When DWQT is reached, the datasets with oldest updated pages are written asynchronously until the ratio goes below the threshold. Here again the writes are always 128 pages.

It is interesting why this limit of 128 is being set?

If we write 128 pages in a forward sequential manner, disk access delays can be minimal. This ofcourse stems from the old disk systems where rotational delay can be a factor. Basically the write can be done without even moving the arm. With modern systems, especially with fast writes, this may not be a big concern. But the major concern is the I/O rate.

Each Cylinder comprises of 15 tracks. Each track in the olden days can hold 10 VSAM Control intervals (DB2 pages). So, technically we can write upto 150 pages without moving the arm. This internal value is indeed specified in ZPARM as SPRMDQ. You can change it, but the option is not available through install CLIST. The change has to be done manually to DSN6SPRM.

Another sub-system event that can cause Asynchronous Write is the Checkpoint. At checkpoint time, DB2 will externalize all those pages for which there is WRITE INTENT. Checkpoint is just a regular system event, which apart from externalizing the pages the checkpoint process updates the BSDS with the Checkpoint RBA, denoting a system wide consistent point used in restart and recovery. As many believe, there is nothing that stops in DB2 during the checkpoint. Checkpoints can be disruptive if many pages wait to be written asynchronously. That is why it is necessary to control the writes with the thresholds. I have writtent a separate document about checkpoint itself.

It is relatively simple to plot the number of pages in the deferred write queue against time so we can graphically represent the I/O burst at checkpoint time.

By scheduling the writes more often, we can smooth out the ripple effect or harmonics. So the goal should be to replace the bigger less frequent writes with more frequent little writes. The other interesting point is during a reference to the updated page. If a page that is in the updated queue is referenced again, it will be removed from the queue, then the write will be completed through a synchronous write. By doing this the read need not wait for the write to complete unless it is actually in the process of being written. It can be severely disruptive, when several pages are infact waiting to be written.

Lowering the VDWQT and DWQT should be tailored to the Virtual Buffer Pool size. If the BPOOL size is 25000, even 1% means, 250 pages. In this case reducing the VDWQT will not do anything. It will still write 128 pages. By doing so, we have just increased the frequency of the writes. So the key here is to arrive at a number that would write less number of pages more frequently. It can be seen that increasing the number of pages above 128 is actually disruptive.

Lowering VDWQT all the way to zero will cause the writes to be triggered when 40 pages pertaining to a single dataset are 'IN USE'. This will trigger Async Writes of upto 32 pages. Lowering VDWQT to zero will have an impact on the CPU to DBM address space. There have been reported situations, where we had set VDWQT to ZERO accidentally in all the bufferpools. We saw huge spikes in DBM1 CPU%.

Before trying to adjust these values, you need to know the behavior of the pagesets in the pool. It is recommended to specify a high value for Sort work VBPOOL because of the way SORT is performed. For most of the application pagesets you may need a low value depending on the page updates.

While it is very difficult to recommend a value for these thresholds without knowing the static and dynamic behavior of the I/O sub-system, in general I always set a value so that close to 64 pages will be written asynchronously when VDWQT and for DWQT, set a value so that 150 pages will be written out when DWQT is hit. While this worked as a starting point, often it is necessary to readjust these values to tailor to the workload. If the number of updated pages is so high, and the effective number of pages written per second is too low, then zero can be considered. Starting from Version 6.0 of DB2 you can optionally specify the no of pages to DEQUEUE when the VDWQT percentage is 0. (VDWQT (0,64))

One of the indicators that would indicate that the deferred write thresholds should be changed is the average number of pages written per I/O. The other indicator is a poor response time from a Synchronous I/O. If the pages written per I/O are in single digits, it is a good idea to adjust (lower) the VDWQT and DWQT values.

Writing data more frequently will also help the NVS in the storage controllers. It can help reduce the restart time of the sub-system following an abnormal termination. During restart DB2 does not have to wait for the status rebuild and write processing for the pending pages. It can be seen that there are many advantages in right sizing these values.