The Good, The Bad, and The Dirty Calc

Executing calculations that only run on blocks that have changed is a great feature in Essbase. It enables administrators to calculate the database in a fraction of the time and is referred to as calculating "dirty" blocks, or an update calc. This is awesome. "Why shouldn't I use it all the time?" you might ask. Understanding how the Essbase calc engine works is critical to answering this question.

The Essbase calc engine calculates each block in a specific order (see figure 1). The first block it calculates is the first level 0 block of the first sparse dimension. It then traverses to higher levels and moves through the dimension from top to bottom until the entire dimension is consolidated.

When a level 0 block is changed, it and all of its parents, are tagged as dirty (it needs to be calculated again). When a calculation is executed on just dirty blocks, the process is the same except that it skips all the "clean" blocks. Once the block is calculated the dirty tag is changed to clean.

So far, so good!

Revisit figure 1, which is a very simple example. It shows a very simple hierarchy with the order in which the blocks are calculated, 1 through 10.

Figure 2 shows what happens if New York is updated. Blocks 5, 6, and 10 are tagged as dirty. The next calculation, if set to calculate only the dirty blocks, would only calculate blocks 5, 6, and 10, in that order.

Here is where things get a little ugly. When an application has write access, as a planning or forecasting application would, it is very possible that users are updating data DURING the calculation process. The timing of these events is critical to understand why calculating only dirty blocks can cause inconsistencies.

When a calculation has started, it identifies which blocks need calculated (5, 6, and 10 in this example). Immediately after that, it starts calculating block 5. If Texas is updated while block 5 is being calculated, what happens?

Figure 3 shows the state of the clean/dirty blocks when the calculation is finished with block 5. It is exactly what you might expect at this point. Blocks 6 and 10 are still dirty. The update of Texas caused Blocks 1, 3, and 10 to be tagged dirty.

This is the critical piece. Keep in mind how the calculation engine works. It will continue to calculate blocks 6 and 10. Also note that the calculation running does NOT reevaluate what needs calculated. It will not calculate blocks 1 and 3.

Figure 4 shows the state of the blocks after the calculation finishes. Only blocks 1 and 3 are dirty at this point because 10 was included in the calculation.

When the next calculation is executed, the only blocks that are dirty are 1 and 3. Can you see the problem now? After blocks 1 and 3 are calculated, is block 10 accurate? Does U.S. equal the total of South, East, and West? Unfortunately, it does not.

One could argue that it will get updated the next time data is changed. In a very simple example with 3 levels, this would probably correct itself rather quickly, if the problem happened at all. In a more realistic example where a company has 10 or 20 levels in their organization dimension, the problem is likely to be a reoccurring problem and may not be corrected until a full calculation is executed. In most situations, it is not acceptable to have a database where it consolidates correctly only some of the time without any warning that it is not accurate. Reporting can be incorrect, and bad management decisions can result.

Using the dirty calc feature is a great tool to have in your arsenal. It can save hours of processing time. It can make you look like a genius. Without understanding its pitfalls, it can be the source of countless wasted hours trying to figure out why a cube isn’t consolidating correctly. A worst case scenario is when a cost center manager updated their budget, it never gets consolidated correctly, and the problem isn’t identified until it is too late.