

PeopleSoft Online Performance tuning Mantras

Tuning is an art; Performance tuning PeopleSoft online workload is more than an art. There exists a rule of thumb. Review the SQL trace and concentrate on those SQL statements that take more than 2.0 seconds to execute. Well, what if the trace doesn't show anything more than 2.0 seconds. Where do we turn to improve performance?

Common reasons for poor online performance

- 1.0 Missing indexes for proper optimization
- 2.0 View materialization (Views with DISTINCT or GROUP BY clause)
- 3.0 Incorrect JOIN predicates
- 4.0 In efficient SQL code
- 5.0 Improper JOIN methods chosen by the SQL optimizer
- 6.0 High I/O rate
- 7.0 In efficient Peoplecode logic

I have tried to arrange the reasons for poor performance as mentioned above in the order of increasing complexity. Online SQL statements are usually very straightforward except a few search views where the view definition may actually hide the complexity of the SQL statement. In most cases it either uses the index or it does not. It is also fairly easy to spot most of the issues. Problems can go out of control when you realize that the performance is poor but you can't get to the apparent cause.

Biggest myth

One of the biggest myths is to consider a CPU upgrade to resolve the performance issues. Before you upgrade the CPU you need to justify the reason first. If you upgrade the CPU while the I/O is the main reason for poor performance, you will just exacerbate the issues for the I/O system.

Just three steps

The first step of online tuning process is to review the SQL trace for the transaction and identify those that need improvement.

The second step is to identify repeating SQL statements. How can we reduce the I/O rate?

The third step is the stress-testing phase. Several problems that don't appear in the SQL trace review as above don't manifest until you stress test the transaction.

Review the number of I/Os; I/O waits for SQL, average CPU for SQL, and average elapsed time for SQL, the percentage of I/O wait in the average elapsed time and the percentage of all known wait-times in the elapsed time. These are key metrics that you need to be concerned.

In terms of percentage, how much CPU you get. For those familiar with the mainframe terminology, it is CPU using as reported by RMF in the sampling interval. Well we don't want to get to that detail. But at least we can determine if we meet our workload objectives.

One of the very important parameter is the I/O rate. If you save 50 I/Os per second you will be saving (50 X 60 X 60 X 8) 1440000 I/Os over an 8-hour period. That is a lot of savings. Who doesn't want that?

When you remove the bottlenecks the CPU will actually drive well. You will get higher throughput. You need not consider a processor upgrade.

I/O rate directly relates to the Buffer Pool efficiency. Contrary to what people believe, online workload responds very neatly to buffer pool tuning. Remember, you cannot reduce the getpages if the panels let your users to do a search on COUNTRY_TBL, 100 times (a little exaggeration here) within a transaction. You can reduce the getpages from becoming a physical I/O.

With DB2 internal traces it is really possible to nail down the hard hitters.

Just three reports

We only need the following three reports.

- 1.0 Information about the workload and a SQL trace (Trace option 7)
- 2.0 DB2PM accounting long or equivalent report
- 3.0 RMF workload activity report

For ORACLE, we need the following three reports.

- 1.0 Information about the workload and a SQL trace (Trace option 7)
- 2.0 UTLBSTAT & UTLESTAT output
- 3.0 SAR or VMSTAT showing CPU, working set and paging statistics

If you want to discuss how you can improve your online response time talk directly to our principal consultant by calling 866-DB2-PSADMIN for further information He can also be reached at venkat@hewittandlarsen.com.