Predicting Multi-Block Read Call Sizes

Jeff Holt, *Hotsos LLC*

Full-table scans don't always read *db_file_multiblock_read_count* blocks at a time. Sometimes an optimization specialist will choose an Oracle full-table scan execution plan over an index range scan to exploit the performance advantage of multi-block reads.

The rationale is simple: head positioning time is the most time-consuming step of a disk read, dominating an Oracle block's worth of data transfer time by as much as 20-to-1. Hence, a multi-block read consumes less read time per byte than a single-block read. So for queries that read a lot of blocks, the full-scan access path is sometimes faster than the corresponding index range-scan access path. But full scans don't always read as many blocks per read call as you might expect.

Example: The trace output shown in Listing 1 depicts several *db file scattered read* events (lines 10–15), each of which represents a multi-block I/O call executed by a full-table scan. For this event type, the parameters *p1* and *p2* denote the file number and beginning block number of the read, and *p3* denotes the number of blocks transferred in each read call. Although the *db_file_multiblock_read_count* parameter for this instance is set to 16, the average multi-block read size for this operation is only 6.3 blocks. There is only one 16-block read call shown in the trace output (line 12). The remaining multi-block reads are smaller than 16 blocks.

The instance parameter *db_file_multiblock_read_count* help defines the maximum number of blocks that Oracle will read in a single full-scan I/O call, but several factors can further reduce your actual I/O sizes:

- **The I/O system beneath Oracle limits the maximum size of a disk read call.** Try setting *db_file_multiblock_read_count* to 1000, and you will see one way in which you're restricted.

- **Oracle will not issue a read call that crosses an extent boundary.** If your extents are very large, then you will have fewer extent boundaries in your database, and your read calls will infrequently be limited by this constraint.

- **Oracle will not issue a physical read call for a block if an appropriate version of that block is already in the database buffer cache.** This behavior often results in smaller multi-block read requests than you might expect.

Example: Oracle's first multi-block read call in Listing 1 (line 10) is a 9-block read of file 12 blocks 71980–71988. Oracle does not show physical reads on blocks 71989, 71990, and 71991, indicating that these blocks were already available in the database buffer cache. Next (line 11), Oracle issues a 4-block read of file 12 blocks 71992–71995, and so on.

If you find that your actual average multi-block read size for your query is significantly less than *db_file_multiblock_read_count*, then the full-table scan execution plan is less desirable than you might have theoretically assumed.

Jeff Holt is a system performance specialist at Hotsos LLC in Southlake, Texas. Prior to joining Hotsos, Mr. Holt was a senior principal consultant at Oracle Corporation, where he diagnosed and repaired performance problems internationally at dozens of Fortune 100 companies.

Listing 1

```
1.  PARSING IN CURSOR #47 len=47 dep=0 uid=12 oct=3 lid=288 tim=26821292 hv=1876399434 ad='86dd7e4c'
2.  SELECT LOC_ID FROM LOCATIONS WHERE REGION = :b1
3.  END OF STMT
4.  PARSE #47:c=0,e=0,p=0,cr=0,cu=0,mis=0,r=0,dep=0,og=4,tim=26821292
5.  EXEC #47:c=1,e=1,p=0,cr=0,cu=0,mis=0,r=0,dep=0,og=4,tim=26821294
6.  WAIT #47: nam='db file sequential read' ela= 0 p1=12 p2=71835 p3=1
7.  FETCH #47:c=8,e=8,p=1,cr=45,cu=2,mis=0,r=1,dep=0,og=4,tim=26821302
8.  WAIT #47: nam='db file sequential read' ela= 0 p1=12 p2=71888 p3=1
9.  WAIT #47: nam='db file sequential read' ela= 0 p1=12 p2=71951 p3=1
10. WAIT #47: nam='db file scattered read' ela= 0 p1=12 p2=71980 p3=9
11. WAIT #47: nam='db file scattered read' ela= 0 p1=12 p2=71992 p3=4
12. WAIT #47: nam='db file scattered read' ela= 0 p1=12 p2=71997 p3=16
13. WAIT #47: nam='db file scattered read' ela= 0 p1=12 p2=72013 p3=4
14. WAIT #47: nam='db file scattered read' ela= 0 p1=12 p2=72018 p3=3
15. WAIT #47: nam='db file scattered read' ela= 0 p1=12 p2=6857 p3=2
16. WAIT #47: nam='db file sequential read' ela= 0 p1=12 p2=6861 p3=1
17. FETCH #47:c=27,e=27,p=41,cr=160,cu=0,mis=0,r=0,dep=0,og=4,tim=26821329
18. STAT #47 id=1 cnt=16727 pid=0 pos=0 obj=30167 op='TABLE ACCESS FULL LOCATIONS '
```