Oracle Automatic Parallel Execution

Joel Goodman
Oracle University EMEA

Keywords:

Parallel Execution, PDDL, PDML, PQ, Statement Queuing, In-Memory Parallel Query

Introduction

This presentation discusses Automatic Parallel Execution, first introduced in Oracle 11g Release 2 and enhanced in subsequent releases. It covers the administration and effects of Automatic Parallelism parameters, Automatic Parallelism behaviour, Parallel Statement queuing, the use of The Resource Manager to control per consumer group queuing and In-Memory Parallel Query.

The need for better management of Parallel Execution Environments

Administrators have long struggled to manage the pool of Parallel Execution servers in their database instances. Defining a default degree of parallelism for an object in the Data Dictionary creates a "one size fits all" solution, which may result in over allocation of execution servers or an under-allocation, depending on the selectivity of any SQL statement predicates. Parallel hints on the other hand may work initially and be appropriate for a specific statement, but may not remain appropriate as the data skew or volume or both change over time.

The need for Automated Queuing

Another problem with manual Parallel Execution is that a statement may be unable to obtain its requested number of execution servers and may be downgraded to have fewer servers or run serially, causing greater elapsed time for batch work or DSS queries. It might be better to wait a short while, and obtain the required number of servers, but unless the developer writes their code to queue the statement in these cases, it will not occur.

The need for In-Memory Parallel Query

When a SQL statement runs in a RAC environment, Cache Fusion transfers whole block images over the interconnect. Since Parallel Query servers transfer results, which contain less data than the block images containing the results, it might improve performance if this were done generally on suitable tables.

Automatic Parallel Execution

Oracle Automatic Parallel Execution aims to solve the problems listed resulting in better use of Parallel Execution servers, improvement in bandwidth of parallelism for processes that really need it,

orderly and automated Parallel Statement queuing when the server pool reaches an administrator defined usage threshold, and use of Parallel Query instead of Cache Fusion in RAC for suitable tables.

Automatic Parallel Execution Details

There are four possible modes for Parallel Execution in Oracle 11g and 12c.

Manual Mode – Which means that no qutomation is done and parallel execution is administered manually as has been done in past releases. In this mode there is no automatic calculation of parallelism, no automatic statement queuing and no in-memory parallel query.

Limited Mode – Which means that automatic degree of parallelism is calculated only for statements accessing objects which have requested the default degree of parallelism; but not for objects that specify a specific degree of parallelism in the data dictionary. In this mode there is no automatic calculation of parallelism, no automatic statement queuing and no in-memory parallel query.

Automatic Mode – Which means that that automatic degree of parallelism is calculated for all statements accessing objects regardless of the degree of parallelism specified in the data dictionary. The calculation is done at hard parse time and then is used as long as the statement is in the library cache. This mode also enables automatic statement queuing and in-memory parallel query.

Adaptive Mode – Which means that that automatic degree of parallelism is calculated for all statements accessing objects regardless of the degree of parallelism specified in the data dictionary. The calculation is done at hard parse time and then is used as long as the statement is in the library cache. But in this mode a feedback loop checks to see if subsequent uses of the statement perform as expected. If not, then the statement is flagged, forcing a reparse the next time it is used, in order to recalculate the automatic degree of parallelism. This mode also enables automatic statement queuing and in-memory parallel query.

Automatic Statement Queuing

In **Automatic** and **Adaptive** modes, Oracle can automate the queuing of statements when the number of parallel execution servers in use reaches an administrator specified threshold. In these cases the statement is queued until other parallel executions end and release their severs so that the threshold is greater than the number of servers in use. At this time, the processes that are waiting to get parallel execution servers, will be given them in a prioritised fashion based on settings for consumer groups in the Database Resource Manager. In this way the application developer is not required to write special code to handle the situation, and queue the request in their application programme.

In Memory Parallel Query

In **Automatic** and **Adaptive** modes, Oracle can use parallel execution as an alternative to cache fusion in RAC databases when tables are of an appropriate size and accessed enough to make this a good strategy. In parallel query, the servers only return results, whereas in cache fusion entire block images are sent over the interconnect. When a suitable table is found by Oracle, then ranges of blocks are associated with specific instances and the block images for those blocks are stored only in their associated instance's buffer cache. This has the effect of distributing the table's blocks and at the same time, it avoids duplicating block images for the same block, in different buffer caches, When the table is accessed in parallel, then parallel execution servers in each instance access the block images that are associated to their insance and return only the results, to the coordinator process. This has the effect of reducing traffic over the interconnect.

The net result is less duplication of block images, a larger effective aggregated buffer cache and less interconnect traffic.

Summary

Automatic Parallel Execution is a collection of several features that reduce administrative overhead; that allow the optimiser to help determine the best degree of parallelism; that save developers the effort of writing code to handle queuing; and which allow better use of the buffer cache and interconnect bandwidth.

Contact address:

Joel Goodman
Oracle University EMEA
The Helicon, 1 South Place
London
EC2M 2RB
United Kingdom

Phone: +44(0)20-78167607
Email Joel.Goodman@oracle.com
Internet: http://DBATRAIN.WORDPRESS.COM