

A structured approach to the process of Enterprise Storage configuration

Approach

There are 5 main areas to consider in the process for optimising the performance of a production quality database:

1. **Physical layout.**

What's in the box.
Number and type of disk drives, LUNs
RAID level
Cache amount and configuration
Interfaces to/from the outside world

2. **Logical layout**

Logical volumes
Volume groups
Stripe size

3. **Filesystem**

Type
Size
Placement (on volumes)
Inodes
Free space overheads

4. **Oracle**

Multiblock read size
Readahead
Oracle cache sizes
Location & size of tables, indexes, rollbacks
Redo log size and placement
Archive logs
Control file number and placement

5. **Tablespaces**

Space management attributes
Extent size / growth parameters

In addition, there are individual project dependent requirements that will cover the topics of

- Planned Growth
- Availability
- Backup/restore requirements
- System and Database administration
- DR
- Interfaces to other systems (including the need for mirrored data)

The first area, the hardware that is available, sets the limits of the absolute best performance available from a given storage solution. The next two areas need to be defined once the key design decisions from the implementation (areas 4 and 5, plus the project dependant attributes) have been made.

In the case of the C/1 reconfiguration work, we already have a lot of information on how the databases behave in real life. This allows us to make some informed decisions to balance hardware limitations against the requirements of the application.

Improvements

From the system monitoring we have been doing for the past 3 years, we know there is a huge I-O bottleneck that limits the performance of the databases – especially during the overnight batch. For example, on Friday 9 June, one of the database servers, executed 272,000 seconds of work on its 20 CPUs. The same day it waited 1,686,000 seconds for disk I-O operations. This ratio of 6 seconds waiting for disks for every 1 second spent processing data is typical of the past performance of both database servers.

By improving the performance of the disk array, batch operations would be improved directly. For example halving the access times of these disks from the current average of 13mSec to a more reasonable 6.5mSec would reduce average batch time from 20 hours to less than 12.

Strategy

There are two routes to improving the performance of a disk subsystem: speeding up access to data and only retrieving data that is needed. The first is an exercise in bottleneck removal and optimising the placement of data. The second is to ensure a match between the configuration of the IO subsystem and the requests that the database puts on it.

The situation we have has two fixed points – the hardware and the database schema. The goal is to work within these limits and reconfigure the variables we can control.

The first stage would involve a review of the high-traffic database tables and indexes. We have information available on the I-O rates of individual database files. This can be used to determine which database objects should be given storage elements of their own (i.e. their own independent disks or RAID groups) and which low usage tables can be located on shared disks.

Once this determination has been done, the next action is to create a layout that supports accepted best practice. Given that this will not be attainable with the existing hardware, we will then need to go through a stage to come up with the best fit, given our constraints.

The second means of improving I-O performance is to only retrieve (or write back) data that is actually required. In practice this means that the I-O parameters that Oracle uses should be matched to the configured hardware. This is especially important for the second and third areas (logical layout and file system configuration) defined above.

Role of Dalleon Systems

We have collected a large volume of database performance information. It can be used to identify which are the important bottlenecks in the current implementation and also (by comparing “good” days with bad ones) what the likely effects of certain compromises will be.

This data will be used to identify which database tables support high rates of access – and when these occur. This will enable you to implement the above strategy from a position of knowledge.

One aspect of the application that is somewhat arbitrary is the number of streams that most of the batch processes initiate. We will be able to use the performance information collected from past batch runs to optimise the number of streams, maybe changing the defaults of either 25 or 50 streams to a more analytically derived (and faster) number.

Finally, we will be able to use database growth information to recommend database entity sizes that are much more resilient to growth. This should improve reliability by removing the need for DBAs to intervene whenever a database entity runs out of space.

Recommendations

The Systems Architecture team is already working with us in the preparation phase of this work. It would be useful to hold an implementation workshop where all the parties with an interest in the 5 stages of optimisation are present.

It would also be useful for the business to provide information of their targets for performance improvement, in quantifiable terms so that this work has some tangible (and pre-agreed) success criteria.

Last of all, I would suggest that this process is used as a template for the design of future storage solutions. As a first run of this structured technique, there are bound to be some improvements that can be made later. It is important that the experience gained from this work is retained and built upon.