

Diagnosing Data Warehouse Performance Bottlenecks

Data warehouses are one method that organisations use to obtain insights into behaviours or activity from a large quantity of data. They are closely related to data-marts and form part of Decision Support Systems. The patterns of behaviour they unearth are the result of long and complex database queries.

Here we look at some of the causes of poor performance with these applications and the options for improving the way they run.

... it's important to know when to stop, as performance improvements can go on until the money runs out.

The first step in any performance diagnosis is to identify exactly who is affected by poor performance. Is it a single user, or everyone who runs a particular application? Does the situation occur all the time, only at certain times or seemingly at random?

To keep things brief and to provide the most relevant information for the largest group of people, we'll focus on the most frequent situations: problems that affect many users, either all the time or during clearly identifiable periods of work. Since Dalleon Systems specialises in problems relating to servers, we'll assume that the users' PCs do not contribute to the problems.

There are 5 main performance attributes to investigate:

- Processor (CPU) usage
- Availability of memory
- Amount and speed of disk access
- Network traffic
- Application (including database) bottlenecks

If possible, it will help if you have this data for periods when the warehouse is giving good service as well as when it's performing poorly. The time interval between samples should also be much shorter than the length of the problem – 5 minutes between samples is probably a good target. It's also worth bearing in mind that a data warehouse may not just be a single server. It may be

segmented. For example into a database plus an application, presentation or web

component, which run on separate servers. You will need a complete overview of the architecture of your platform to make sure no parts of it are overlooked.

Goals

The nature of data warehouse work is that queries tend to be resource intensive and users wait while they are executed. This means that the time taken to deliver a result is set by the speed of the slowest, or the most heavily used component of the system. Our goal is to discover what this is and set out options for improving it's performance. Be aware that as soon as one bottleneck is removed, the performance will become limited by the speed of the next-slowest part. Therefore, it's important to know when to stop, as performance improvements can go on until the money runs out. Success is when the server is delivering results at an acceptable speed.

Strategy

The two simplest attributes to eliminate as bottlenecks are network traffic volumes and the amount of free memory on the server. However, you should be aware that these are also the least likely causes of performance problems.

Network

My preferred option here is to *ping* the data warehouse server and any associated application servers either from an affected user's PC, or somewhere close to it. This will tell you not only if the server itself is having trouble servicing network requests, but also if any other infrastructure in the path to the server is causing problems. If there is any significant variance between ping times when things are working well and during problem periods, they should be investigated.

Memory

The effect of too little memory is that parts of the application have to be continually read from disk in order to be executed. This is a slow operation and *excessive paging* is a sign of insufficient memory. A second area where additional memory improves performance is to increase the size of the database *cache*. Check the cache-hit rates of the database server – if they are below (say) 90%, the increased number of disk operations may be causing poor performance. Before jumping in and blaming lack of memory, check the other attributes for shortages, too.

CPU

Running at 100% processor utilisation is not usually a bad thing for a data warehouse. As discussed above, since warehouse work involves long-running queries,

there will always be one attribute of the server that is used at its maximum capacity. The question here is whether the maximum utilisation is itself due to under-performing processors, or if it's caused by a poorly tuned database, or inefficient SQL. If it's down to the last two, upgrading by changing to faster processors – or adding more, may only be a temporary solution. You should also continue the investigation into the setup of the database and the construction of the queries it's running.

Disk

This is where it starts to get interesting. Here you should focus not simply on the number of disk accesses per second, but the length of time each one takes (on average). Data warehouse applications typically require very large amounts of data to analyse and are therefore very dependent on the speed this can be read from disk. Frequently, data is fetched by JOIN-ing two or more tables – which requires data to be read from multiple locations simultaneously. If all the data is stored on the same disk, each record must be retrieved sequentially. However, if each table is placed on a dedicated disk, each one can be accessed at the same time, thus greatly speeding up the time needed to JOIN the tables. You will need to investigate both the absolute maximum rate of disk access (aim for not more than 100 I-Os per second from each physical device) and also which files or tables are accessed together. Split these tables onto different disks to remove access bottlenecks.

As mentioned in the paragraph about *memory*, make sure that

your database's cache is sufficiently large to satisfy most database requests without having to access the disks for each single record.

Finally, a well structured data warehouse make heavy use of database indexes. Therefore it's vitally important to make sure that not only are the correct tables indexed (see the next section), but that there is no conflict between data accesses for indexes and data tables – or other database files.

Since disks are the only mechanical devices in your server (excluding the fans, but they don't affect performance), they are inevitably the slowest. Consequently every effort should be made to ensure that each device is used as little as possible – both by spreading data across a large number of physical disks, even if you use only a small proportion of their “gigabyte” capacity, and also by ensuring that each one is only used for one thing during a given query.

Database and Application tuning.

This is a specialised field. Apart from basic database monitoring (number of queries, cache efficiency, etc.) you will probably have to get a DBA and/or an someone who wrote the warehouse queries involved. To do this, it's wise to be reasonably confident that there are no major bottlenecks elsewhere, so that these specialists will not be spending their time looking for inefficiencies that do not exist. You should also be careful how you engage these individuals, as some might take it as a personal criticism that you are accusing them of writing poor applications, or

of not doing their job of administering the database.

To avoid these personality issues, I try to have a snapshot of the SQL that is executed when the warehouse is running slow and to compare this to what is executing when things are going well.

External factors

You should also be aware of what other activities are going on around you when a performance problem strikes.

In this case, be aware that data warehouses have to have new data loaded into them, old data purged or summarised and some existing data will be updated. There may well be jobs running to perform these tasks and that conflict with the users' work when they are experiencing poor performance.

Similarly, database backups do not necessarily run with the knowledge or approval of the users. On enterprise class servers, it's frequently the case that disks are mirrored either for flexibility or resilience. One popular technique for backing up a running database is to “break” the mirror and back up the off-lined disks. Once the mirror is re-created, there can be a large backlog of data that needs to be updated onto the mirror volumes. This will not show up as database or even server activity – just as mysterious slowdowns in disk performance.

Dalleon Systems provides services and consultancy for monitoring the performance of your servers. For more information visit our website at www.dalleon.co.uk