Condusiv™ Technologies

WHITE PAPER

Keeping Your SQL Server™ Databases Defragmented with Diskeeper®

All SQL Server databases, over time, experience "internal" fragmentation of their data. This occurs when records are removed from database pages, but the space it occupied is still there after deletion. Eventually, this space is reused, but as it is reused, the data pages become fragmented, which can lead to unnecessary I/O, especially in case of table scans where many data pages are read, one after another.

In SQL Server, there are several ways to defrag internal fragmentation. One of these methods is to use the DBCC REINDEX command to rebuild clustered and non-clustered indexes. Once indexes are rebuilt, data pages are now logically contiguous, and disk I/O is minimized.

Unfortunately, internal fragmentation is only part of the fragmentation problem. When DBCC REINDEX is run, it does nothing about "external" fragmentation. External fragmentation refers to the fragmentation of files on your server's disks, which can cause as much, if not more, unnecessary I/O activity as internal fragmentation. Unnecessary I/O activity, as you would expect, hurts SQL Server's overall performance.

SQL Server databases are made up of large database and log files that are pre-allocated in size at the point of their creation. If there is enough contiguous empty space on disk when the original files are created, then they will not be fragmented. But if the empty space available is not continuous, then these original database and log files will be fragmented.

Even if the original database and log files are not fragmented when they are first created, they will almost certainly become fragmented as the database grows over time. For example, if you set the original database size to 100MB and the log to 10MB, and you have them set to grow automatically, and if eventually the database grows to 5GB in size and the log grows to 100MB in size, external fragmentation could become great. Every time the database or log files grow automatically, there is the potential for external fragmentation.

To defrag external fragmentation takes an operating system utility, not a SQL Server utility. One of the most popular tools for defragmenting SQL Server database files is a tool from Condusiv Technologies Corporation called Diskeeper. Diskeeper has been around for many years, and many of you may already be familiar with it, at least as how it is used for Windows file and print servers. What many DBAs aren't familiar with is that it is the best tool for defragmenting external fragmentation on their SQL Servers.

When an external fragmentation tool like Diskeeper runs, it does not restructure the internal contents of the file, unlike DBCC REINDEX. After Diskeeper defragments a file, the defragmented file will be a bit-for-bit duplicate of the original. Therefore, any holes within the database are still present and from time to time you will still need to rebuild your indexes to combat internal fragmentation.

There are two types of external fragmentation with which a utility like Diskeeper deals with: file fragmentation and free space fragmentation. File fragmentation concerns computer disk files that are not whole, but rather are broken into scattered parts; while free space fragmentation means that the empty space on a disk is broken into scattered parts rather than being collected all in one big empty space. File fragmentation causes problems with accessing data stored in computer disk files, while free space fragmentation may cause problems creating new data files or extending (adding to) old ones.

So when Diskeeper runs, it acts to defrag database and log files so that instead of being made up of many pieces, the file is one continuous segment. In addition, Diskeeper defrags free space so that when database or log files expand, they can expand with little or no fragmentation. But this does not last forever. Eventually, fragmentation becomes a problem, and the database and log files need to be defragged again. Ideally, defragging should be performed on a real-time basis to prevent unnecessary fragmentation from occurring and impacting performance.

Now here is something you probably have not thought of before. What effect does file fragmentation have on rebuilding your SQL Server indexes? In other words, if you don't perform file defragmentation, but you perform an internal index/record defragmentation, are there any downsides to this?

Yes, there can be. Because the files are fragmented, it will take SQL Server much more time, and I/O, to rebuild its indexes on fragmented files than it does on contiguous files. This means that before you perform an internal defragmentation process that you might want to perform a file defragmentation process first. This way, you reduce how long it takes to rebuild your indexes, and you also reduce the I/O load on your server during the index rebuilding process.

While SQL Server database and log file fragmentation can have a negative effect on SQL Server's performance, don't forget that there are other files that SQL Server accesses, such as the SQL Server executables, and if you are using Full-Text Indexing, the full-text index files. So not only do you want to defrag SQL Server database and log files, but all files located on your SQL Server.

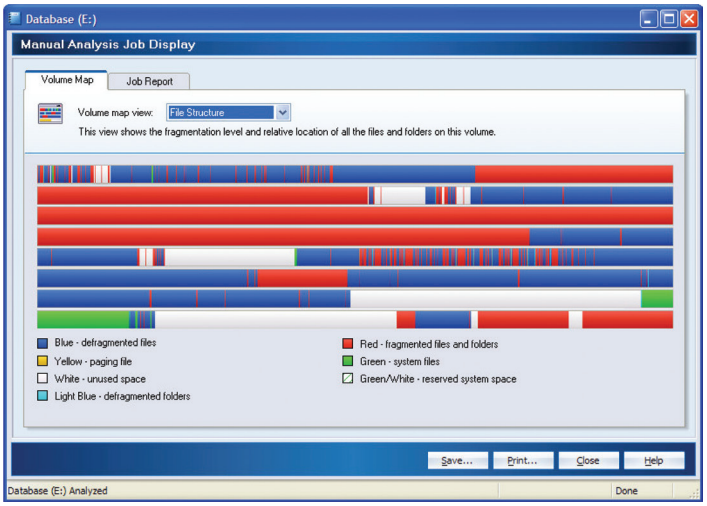

Figure 1.0 - Screenshot of Diskeeper Analysis

All file movement in a Diskeeper defragmentation job is handled by the operating system itself. In fact, the code in the operating system, which was originally co-written by Condusiv Technologies Corporation, prioritizes safety in determining what can be defragmented and what cannot. SQL Servers databases (e.g., .LDF, .MDF) are perfectly safe to defragment. As Diskeeper sends requests to the operating system (through an API) to move files, if it comes across files that cannot be safely moved, they are simply skipped over without any error or concern.

So how do you find out how if the files on your SQL Server are fragmented? Fortunately, this is easy. As part of Diskeeper's functionality, you can run a fragmentation analysis to see just how fragmented your SQL Server files are. As with defragmentation, this can be done while SQL is running.

As you can imagine, it is hard to recommend a specific time frame during which to run file defragmentation, as each database is different and fragmentation occurs at different rates. A dynamic Diskeeper feature defragments on-the-fly using proprietary technology called InvisiTasking™. Diskeeper will only defragment files or free space if it is needed; and with InvisiTasking, it guarantees that defragmentation operates invisibly, with zero resource impact to SQL Server and other running applications. And, yes, you can still restrict Diskeeper's runtimes if you so choose.

Many DBAs professionals who purchase Diskeeper originally begin evaluating the product after they experience issues they determine stem from badly fragmented disks.

It typically starts out that something slows to a crawl. As they investigate they use systems management and monitoring tools to narrow down the source of the issue. This is especially true on SQL servers, where split seconds can add up to thousands of dollars of lost revenue.

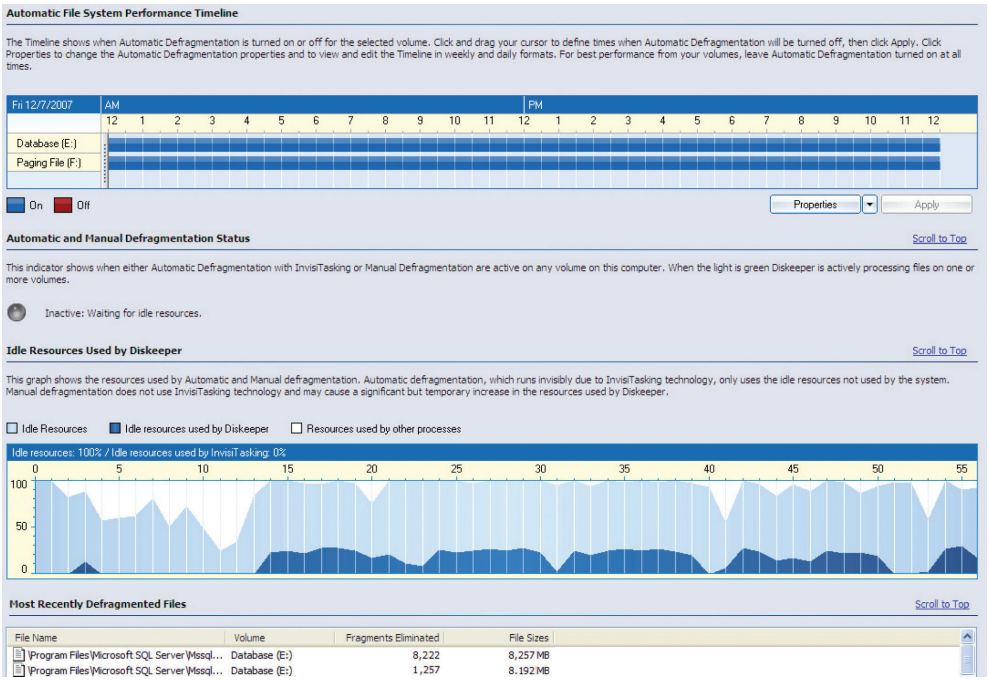

Figure 1.1 - Screenshot of Diskeeper with runtime selection and resource usage display

Numerous technical articles delve into specifics to help diagnose this situation. One such paper on SQL Server is Troubleshooting Performance Problems in SQL Server 2005. In the section I/O Bottlenecks, it describes a number of counters to look for that can help pinpoint the source of the poor performance.

Per the Microsoft article, the following is a gauge of what Average Disk Sec/Read:

Less than 10ms = very good
Between 10-20ms = okay
Between 20-50ms = slow, needs attention
Greater than 50ms = serious I/O bottleneck

Physical Disk	Total
Average Disk sec/Read	212.4

Figure 1.2 – Line item from PerfMon.exe Text Report

Reported cases [to Condusiv Technologies Corporation] from SQL DBAs have shown Average Disk Sec/Read to be exceptionally high on their servers, including delays of 200ms and 300ms, well into the serious I/O bottleneck part of the scale.

As expected, the article provides possible solutions to address I/O bottlenecks. However, one solution not explicitly mentioned is file defragmentation. Defragmentation alone has been shown to reduce Average Disk Sec/Read from the 200-300ms range back to around 15 and 30ms, a tremendous improvement in performance!

It isn't uncommon for defragmentation to be left off a list of solutions, as it is still unknown to many. Recommendations to increase I/O bandwidth, such as adding more disks, are indicators that you need to investigate file fragmentation as a possible cause. Example: your SAN vendor tells you to add another controller or array. If there is substantial fragmentation, Diskeeper will typically provide better results, as it actually fixes the issue rather than camouflaging it.

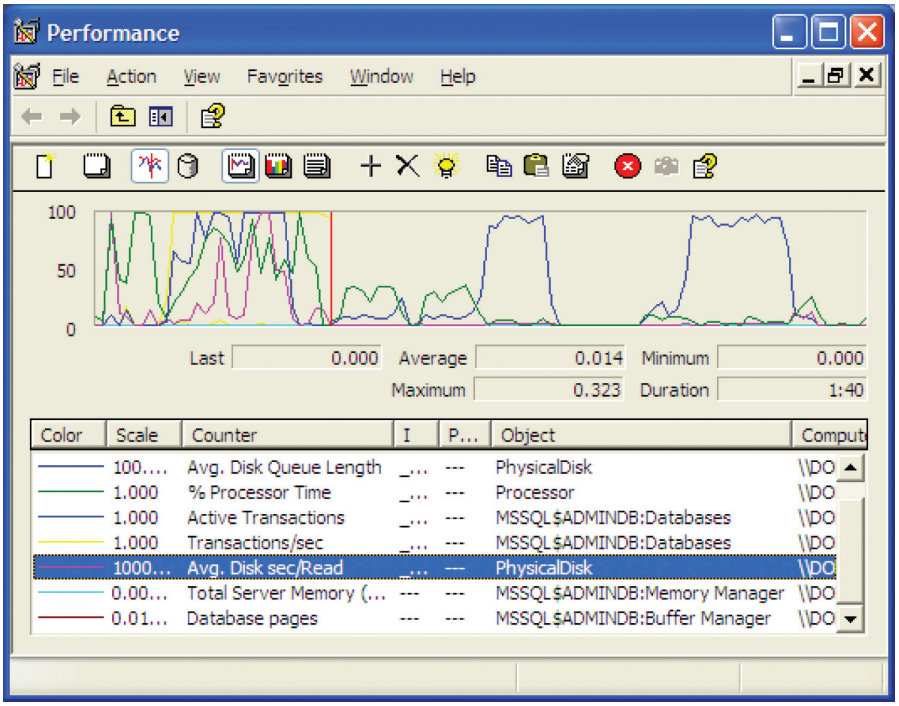

Figure 1.3 – PerfMon.exe Graph Report

In summary a file defragmentation utility like Diskeeper can take care of the external disk fragmentation, while a SQL Server utility like DBCC REINDEX can address internal SQL Server disk fragmentation. As a team, they can work together to help ensure the optimum performance of your SQL Servers.

If there is any doubt, simply install Diskeeper and use its analysis function to find out exactly how many individual pieces your files are broken into. You're likely to be very surprised. It's quite common to see analysis reports with database files fragmented in hundreds of thousands of pieces!