SQL Server Storage Engine Blog

Data compression techniques and trade offs

Sunil Agarwal 30 Sep 2007 2:55 AM 9

Ok, now that we have sort of agreed (http://blogs.msdn.com/sqlserverstorageengine/archive/2007/09/30/data-compression-why-do-we-need-it.aspx) that data compression is a good thing, you may wonder how SQL Server compresses the data, what does this compression mean to my data and to my workload?
If you are a theory nut or crave for mathematics, you can browse the web or read the related books and find many fine techniques to compress the data but note, not all compression techniques are applicable to databases. I finished my engineering degree almost 30 years ago and, thanks to the computer science profession and the hours spent in coding/testing, developing products and now with my work as a program manager, I have lost touch (my excuse) with physics/mathematics, the subjects I used to love the most in college.
In general, there are two kinds of compression as follows

- Lossy: This causes some loss of data when it is uncompressed. This is done to get fast/better compression. It is more suitable for compressing images, music and videos files where some loss in fidelity is undetectable to human eye/ears. One example is JPEG. Clearly, this is not suitable for documents or for databases.
- Lossless: This allows original data to be reconstructed exactly from the compressed format. This is what we need for data in the database. There are two common techniques here. First technique is based on the encoding the input data in such a way that more commonly occurring data is coded using less number of bytes. One example of one such technique is Huffman Coding. Second technique is based on statistical modeling for text data. One example in this category is LZ (Lempel-Ziv) algorithm. There are many versions of it but the central idea is to build a dictionary of common occurring symbols in a block and then reference these using a pointer. So the commonly occurring symbol need only be stored once there by saving the space. SQL Server uses its own proprietary implementation of dictionary based algorithm.

Let us first get two important points out of the way.

1. The goal of compression in database is NOT to get the maximum compression possible but to find a right balance between the compression achieved and the cost of compressing/de-compressing the data. It will serve you no good if SQL Server can compress the data 90% but the CPU cost of SELECT and DML operations become unacceptably high. You often hear that so and so database system compresses the data 60% while the other database system compresses the same data 40%. The important question to ask is what is the impact on the workload (i.e. CPU)? If the impact on the workload is unacceptable, the customers will not use the data compression in spite of better compression. Ideally, we need to strike a right balance between the compression achieved and its CPU impact. For this reason, SQL Server team evaluated various compression techniques, estimated the data compression we could achieve for different DW databases, estimated the CPU impact of compression/decompression and then zeroed-in to the types of compression to deliver what we have in SQL Server 2008. It is not to say that we won't enhance compression in future. We will always look for opportunities to enhance our data compression offering.   So please do always feel free to contact us with your ideas and the kinds of data challenges you are facing.
2. The compression achieved depends on the data distribution and the schema. So for example, if all integer values that appear in a column can fit in 1 byte and you have declared the column type to be INT, you can possibly save 3 bytes (i.e. 75%) barring any row-overhead per column values. On the other hand, if you had declared the column to be tinyint (i.e. 1 byte), then there are no savings with compression. So the same data but different schema will give you different perception about the compression. Similarly, for the data distribution.

In the next BLOG I will provide more details on types of compression in SQL Server 2008

Tweet 0 Like 0 Share Save this on Delicious

Comments

30 Sep 2007 3:04 AM
Techy News Blog » Data compression techniques and trade offs

PingBack from http://www.artofbam.com/wordpress/?p=3966

30 Sep 2007 2:19 PM
SQL Server, BI and .NET

Dal team dello Storage Engine di SQL Server arrivano alcune notizie interessanti a proposito delle nuove

30 Sep 2007 5:17 PM
SQL Server Storage Engine

As announced in Tech-Ed 2007, data compression is a new and exciting feature targeted to be available

15 Feb 2008 10:35 AM
Michael Reinhard

By the way it would be really interesting to hear at first hand about effectiveness curve of using compression for storing various data. It's also interesting how compression impacts encryption. I absolutely agree here with you about the importance of compression for backup operations. I used to use scripts to apply external compression to SQL backup files before I copied them to storage, but that solution performed very poorly. Furthermore, such a solution usually prevents you from running fast and seamless operations on backup files because you have to involve a stand-alone decompressor and a script before performing every operation on that backup file you have created using this scheme. That was one of the reasons why we purchased Scriptlogic's LiteSpeed http://scriptlogic.com/products/litespeed/ . That seriously increased our performance thanks to a fantastic SQL 2005 database engine and techniques implemented in Litespeed. You needn't go far to get some examples. I've just backed up a mid-size database of around 1.5 gigs on SQL Express 2005 where it achieved a compression ratio of 77%. I am keen to see SQL Server 2008 in my environment. I think that it's one of those best technology cases where a pair of solutions from different vendors make a burning mixture that does the job the best way you'd have preferred. What I like in such tools like Litespeed is the ease of procedures you can perform using the intuitive and effectively built configuration controls. It's like with SQL Express' 2005 setup wizard. You need to select a couple of settings and you are done with the setup. It really was a surprise to me when I saw that a backup operation takes shorter time if you do it by applying compression on the fly. The high CPU speed of modern systems really does the trick. In your very first article you've told that "Well, if the data is compressed, you can fit more data in the same memory." Isn't it effective only in the case of asymmetric algorithms without? By the way, you said that the data is decompressed by row. That is it's JIT compressed. Do you use caching to somehow analyze the frequency of fetching rows? Or the mechanism works similarly to deflate implemented with NTFS compression? Does it have any problems with compressing large files http://support.microsoft.com/kb/927912 like NTFS does? If it is practically impossible I would be really great to be able to effectively use a combination of built-in SQL server compression and the compression applied with Litespeed. I've seen such problems with compressed NTFS files many times.That's why I decided that I leave production system uncompressed and apply compression only to the DBs that I backup.

18 Feb 2008 3:38 PM
Sunil Agarwal

Thanks for your comment. Some asnwers here

(1) if data is encrpted before comrpession, you will not get much comrpession. TDE (a new featrure in SQL2008) encrupts after compression so you can have both compression and encryption

(2) SQL Native backup compression is a very competitive offering. I suggest that you compare with lite-speerd. Other point that I want to make is that backup is typically IO bound, So if you use data comrpession and say you are able to compress 50%, then your backup will finish in 1/2 the time.

(3) Data is kept compressed in the buffer pool and we 'decompress' only the requested columns, not the entire row or page. We don't cache 'decompressed data at this time. I did not understand your point on "Isn't it effective only in the case of asymmetric algorithms without"

(4) The compression/decompression happens at column/row/page level. The size of the file only matters because there are lot more pages to compresss but the effort is linear with respect to the number of pages. I recommend you try data compression and see the impact on your workload.

25 Feb 2008 11:56 PM
kanchangehlot

Nice to read to such a nice thing in easy-to-understand language.I amn doing

a short project on information and entropy and I am interested in knowing

factors which limit the extent to compress data without any loss of information and

secondly how images are compressed under JPEG. Well ,I would like to tell you

That I am a Physics student , so please don't use engineering jargons.

26 Feb 2008 12:06 PM

Sunil Agarwal

I would suggest looking at the web. I found the following links.

http://www.prepressure.com/library/compression algorithms/jpeg

http://www.jpeg.org/jpeg/jpegls.html?langsel=en

thanks

22 Apr 2009 8:18 AM

RajeshEMC

Hi Sunil,

I saw your articles ,it's Awesome .

I am facing some issue in data compression , My client having around 20 TB of data in SQL Server . The client need to do data Compression on this and need to make a SSAS Cube for SQL Reporting.

I just want to know where can have compression of Data's , It can be in SSAS level or SQL Server Database Level .

If it SSAS Level : How and What are the step should be consider ?

If it SQL Server Level : What the better solution for 20 TB Data Compression ?

NB: The entire data store into a single Table in SQL 2005.

Please help to achieve this.

Thanks,

Rajesh

23 Apr 2009 6:33 PM

Sunil Agarwal

I saw your articles ,it's Awesome .

[sunila] thanks

I am facing some issue in data compression , My client having around 20 TB of data in SQL Server . The client need to do data Compression on this and need to make a SSAS Cube for SQL Reporting.

I just want to know where can have compression of Data's , It can be in SSAS level

[sunila] the data compression I described in not in SSAS...I am not sure if SSAS has its own compression

or SQL Server Database Level .

If it SSAS Level : How and What are the step should be consider ?

If it SQL Server Level : What the better solution for 20 TB Data Compression ?

[sunila] please look at the compression strategy part of the blog...if you have read-mostly data, we recommend PAGE compression

NB: The entire data store into a single Table in SQL 2005.

[sunila] not sure what you mean..looks like you have one dominant table...this is definitely a manageability challenge...Have you partitioned it?