Technology Review

Trevor Hammock
CS 461: Fall
Group: 18
HP Data Compression
Role: Researcher

November 28, 2017

Abstract

HP's PageWide Web Press division develops and troubleshoots their industrial web presses. These web presses send over 350GB a day of business analytics and product issues. This has caused a storage/performance dilemma for their team's back-end database. To solve this my group will research and test various compression options available for Oracle Databases. For this project, as the researcher, I intend to find existing solutions that will be easy to implement into our client's current product stack. Once these solutions are found, they will be analyzed using preexisting research combined with some empirical analysis.

This paper will look into three separate pieces. The first will be System Level Compression which is enabling compression at the physical level. Due to licensing and the major undertaking it would require to migrate to a different environment, this first piece is considered out of scope for this project (but still a valid solution). Next we have Index Compression which compresses the indexes in a database to reduce the amount of space that they occupy. Finally we have Backup Compression which is compressing the backups so that they take up less space on the system. Since backups tend to be compressed to begin with and the location of the backup may not affect the main system, it is not an ideal solution but it will still be looked into anyway.

Contents

1 System Level Compression

1.1 Overview

System Level Compression is when the physical level, which is typically the operating system or the hardware, compresses or decompresses the data on the fly. There are a couple of advantages with this approach with the first being that any application or product that interfaces with that data can seamlessly use the data as if it is not compressed. Another advantage with this approach is that the system performing the compression can apply space saving techniques at the block level. This means that different types of data can be compressed if they have any similarities at the byte level.

The main System Level Compression I will be looking into for this is ZFS. ZFS is a special enterprise file system that serves as both a file system and a logical volume manager. This unique property allows the file system to keep track of the physical mediums and the individual data stored within its volumes. This property is the main aspect of the ZFS file system and allows for features such as the protection against data corruption, efficient data compression, and software level RAID as a few examples.

1.2 Criteria

Given the benefits and the convenience of System Level Compression, this could be an idea solution that could significantly reduce the size of my client's database and all of the data within it. However, trying to migrate or integrate this into ones current production environment will require considerable amount of planning and testing. Because of this, this section is considered out of scope for my project and will not be looked into. The research and the solutions that do come out of this section are not completely irrelevant and could maybe serve as the foundation for a future project. As such only solutions that can easily be obtained from a pre-configured ZFS will be looked into.

1.3 Potential Choices

1.3.1 ZFS Deduplication

One of the features of the ZFS file system is Deduplication. Deduplication is a background process that prevents duplicate copies of data from being stored. It does this by using a deduplication table which keeps track of data by using its checksum. If data already exists within the table, then it points that duplicate data to where it already exists on disk and increments the reference count. Assuming one has a dataset that has a lot of duplicate data, then this feature can provide enormous space savings.

However in practice ZFS's Deduplication is normally discourage from being used. There are a couple of reasons for this but the first is that the deduplication table is very RAM intensive and tends to use anywhere from 3 to 5 GB of RAM per TB of storage. Obviously if one has a lot of storage and/or does not have an adequate amount of RAM, then the deduplication table can get paged out which can severely hinder the performance of the system. The second problem with Deduplication is that data is more likely to be similar than identical. Basically Deduplication only works when the data is identical to each other. So unless one is working under a set of conditions where they can guarantee that a large percentage of the data is duplicated, then it is likely to get outperformed by regular compression.

1.3.2 ZFS Compression

Another feature of the ZFS file system is dataset Compression. Compression is also a background process that can reduce the overall size of the data that gets stored into the physical medium. Since data needs to be compressed when written to and decompressed when read, this incurs a CPU performance penalty. The compressed size of the data and the performance penalty which comes with it all depend upon the algorithms that are used.

Typically there are multitude of algorithms that can be provided with a ZFS file system but I will briefly describe the four common ones. One is ZLE which is one of the fastest algorithms but tends to get the worst compression ratios. Second we have GZIP which can have various levels, like 1 though 9, where higher levels give better compression ratios at the cost of performance. Third we have LZJB which provides a decent data compression ratio but is considered depreciated. Next we have LZ4 which provides better compression ratios than LZJB on top of being much faster, replacing the latter.

1.3.3 Oracle HCC

Assuming you have a Oracle Database instance installed on a supported ZFS file system, one can enable a Oracle's Hybrid Columnar Compression (HCC). Oracle's HCC is a special compression feature that is optimized for their Databases. Using ZFS's compression capabilities, it stores data in a unique combination of column and row formats that help maximize compression and performance.

Currently as of Oracle 12c when this document was written, there are currently four types of compression algorithms that can be applied on the data. The first is "Query Low" which uses the LZ0 compression algorithm which is the fastest algorithm that Oracle offers (but has the worst compression ratios). The second is "Query High" which uses the ZLIB compression algorithm which is a little bit slower than the latter but tends to get better compression ratios. These two options are recommended by Oracle when one wants to optimize more for query speed.

The third is "Archive Low" which uses a higher level of the ZLIB compression algorithm that tends to get much better compression ratios at the cost of speed. Finally we have "Archive High" which uses the BZIP2 compression algorithm to get the best compression ratio but requires a lot of CPU horse power to compress/decompress the data. These last two options are recommended by Oracle when one wants to optimize for space savings and are in an environment where the data is rarely accessed.

1.4 Discussion

In general, ZFS Deduplication is a nice feature but unless one has dataset with lots of duplicate values, then it really should not be used. Next we have ZFS Compression which can compress data on the fly. This tends to perform much better than Deduplication and theoretically be applied on any data on the system. This means that one could try to compress the entire database instance to try to achieve even more space savings but could possible run into some performance issues. Finally we have Oracle's HCC which is optimized for their databases and has many of the same benefits as the regular ZFS Compression. This could integrate well into my client's current Oracle based stack on top of providing better compression ratios on the data itself due it being optimized for this specific type of data.

1.5 Conclusion

In conclusion, System Level Compression can seamlessly compress ones data while making it appear and act as if it is uncompressed to any application. The ZFS file system is a enterprise product that provides a multitude of features such as System Level Compression. This includes data Deduplication and Compression which can reduce the overall storage footprint of that data. This may also Include Oracle's HCC if one is using also running an Oracle Database instance on a supported system. This can also compress the data at the system level but is specifically optimized for databases which can achieve even better space savings.

2 Index Compression

2.1 Overview

In Databases, an index is a special data structure that contains a copy of the selected columns. This allows for very efficient searches and access to the data in those columns. Their are many uses for indexes but some of the typical ones are when one wants to speed up searches on a given column or speed up the joining of tables. However one problem with indexes is that they have a storage penalty associated with them. This is due to the fact that all of the data in that indexed column gets duplicated into a sorted data structure. Causing indexes to "often take up to 50% of the total database space."

2.2 Criteria

As more data is feed into the database, the columns with indexes on them will also grow in size. If one could compress their indexes, then they could significantly reduce the overall storage footprint of the index. Because of this, my teammates and I will look into index compression options as a means to solve our client's storage issues. Options that we will look into should hopefully be supported on there current platform, have a noticeable impact on the overall size of the indexes, and have minimal impact on system performance/maintainability.

2.3 Potential Choices

2.3.1 Oracle's Index Compression

Oracle Database's have supported index compression for a long time which was added all the way back in version 8. Enabling it is simple and currently does not require any licensing. Oracle's Index Compression functions very similarly to their basic table compression. The index is compressed on a per-block basis where duplicate values are stored within a symbol table and replaced with a corresponding token.

However where they differ lies in how they are stored. With Index Compression, the index is stored as two parts which are the prefix and the suffix. The prefix is the leading part of the data and typically is duplicated within that block. The suffix on the other hand is the trailing part of the data that is unique for that given entry. By reusing the prefix and having the suffix serve as the identifier, the data can be easily compressed within that block.

As long as the index is non-unique, Index Compression can reduce its overall size. Additionally Oracle also provides DBA's a way to define what the prefix and suffix are when enabling Index Compression. Allowing the DBA to fine tune how the data will get de-duplicated.

2.3.2 Oracle's Advanced Index Compression

In version 12.1 of Oracle Database, Oracle added a new type of compression called Advanced Index Compression. Advanced Index Compression functions very similarly to the regular Index Compression, however it has one crucial difference. It now automatically chooses an optimal prefix for you on a per-block basis. Basically prefixes that got good compression ratios in one block may be less optimal in another. By choosing the optimal prefix for each block, one is more likely to get the best compression ratio for each block and significantly reduce the overall size.

Additionally Oracle has also added the option to compress partitioned indexes on a partition-by-partition basis. Allowing a DBA to fine tune index performance and size by applying (or not applying) compression on each partition. The following options for this are no compression, regular Index Compression, or Advanced Index Compression.

2.3.3 Oracle's Advanced Index Compression High

In version 12.2 of Oracle Database, Oracle added a new Advanced Index Compression High option. Advanced Index Compression High reuses all of the properties from the Advanced Index Compression but also integrates some previous Oracle features as well. With the new option data is now stored within two sections of the block, the compression unit and the buffering zone. The compression unit is where the compressed data gets stored. However as data is inserted into the block, it gets put into the buffer zone as uncompressed data. Then just like Oracle's Advanced Row Compression, once the block's fullness approaches an internal threshold it will then compress the buffered data. This is done to minimize the compression cost by queuing up the data into a single operation.

Another new feature of the option is how it compresses the data. Just like Oracle's HCC, data is stored into compression units. Once the data is stored into a compression unit, the data is quickly analyzed and applies an appropriate compression algorithm. This allows for more aggressive compression ratios than the typical deduplication one will allow for which greatly increases the amount of data that can be stuffed into that block. This is also done on a per-block basis meaning that each block will be optimized in real time to get the best compression ratios possible.

2.4 Discussion

In general, if one is working with a compatible version of Oracle Database then they should try to avoid Oracle's regular Index Compression. This is due to a multitude of reasons with the first being that indexes with a higher cardinality can actually have negative space savings. On top of that the DBA implementing this needs to be aware of the uniqueness of the columns when choosing the right prefix. Which may also require some additional testing in order to get this right.

Both of the Advanced Index Compression options solve most of these issues on top of achieving far better space savings. With Advanced Index Compression, one does not need to really worry about negative space savings because data is optimally compressed on a per-block basis. If one wants to achieve the greatest space savings, they should probably use Advanced Index Compression High.

However, Advanced Index Compression High might not always be the best choice. It currently requires the newest version of Oracle Database and an appropriate license in order to use it. Plus the higher CPU overhead from the more aggressive compression algorithms could cause some performance and maintainability issues (especially when one may need to rebuild their indexes).

2.5 Conclusion

In conclusion, Oracle currently offers three different compression options for indexes which are Index Compression, Advanced Index Compression, and Advanced Index Compression High. Index Compression is the most widely available but must be fine tuned to get it right. Advanced Index Compression gets far better space savings without any of the hassle of the latter. Finally we have Advanced Index Compression High which can achieve the best space savings but may not be the best option due to licensing and performance issues.

3 Backup Compression

3.1 Overview

In production environments, it is a common practice to make backups of the data or the underlying system itself. This is done in case some sort of disaster occurs such as a hardware failure or a power outage. If one has a recent backup when said disaster occurs, then they can use that backup to restore the data/system to a previous version.

3.2 Criteria

Since backups are done on a routine basis and are rarely ever touched, they are typically compressed. This is usually done with some sort of archival format that greatly maximizes the compression ratio to achieve as much space savings as possible. As a result more aggressive compression algorithms are used which tend to have a much larger impact on system performance and take far longer to complete. This section will look into backup compression as a means of reducing the overall storage footprint of the system. The options that will be looked into must integrate easily into my client's current Oracle based stack, try to maximize the amount of space saved, and try to minimize the impact on system performance.

3.3 Potential Choices

3.3.1 RMAN Basic Compression

RMAN stands for Recovery Manager and is a backup/recovery tool that Oracle has provided with it's databases since version 8. RMAN's primary feature is backing-up Oracle Databases and recovering them by loading said backup. RMAN also provides some cool features such as increment backups which store the changes between the system rather than an entire copy of it, setting up routine backups based on a set of criteria, storing said backups on the same system or another one over the network, and many more. RMAN also provides an option for compressing the backup that is available to all without any licensing. According to Oracle, it uses some sort of binary compression and can achieve some pretty good compression ratios within a moderate amount of time.

3.3.2 RMAN Advanced Compression

Later on when Oracle introduced Advanced Compression to newer versions of their databases, RMAN also inherited some of those features as well. Oracle now offers RMAN Advanced Compression which, assuming you have the Advanced Compression license, can compress your backup now with a range of options. These options are LOW, MEDIUM, and HIGH which will allow you to choose between minimizing system performance to maximizing space savings.

The LOW option is the lightest option available but tends to have less space savings then the basic option but uses a fraction of the system resources. Next we have the MEDIUM option which performs similarly to the basic option but uses less system resources. Finally we have the HIGH option which gets far better compression ratios then the basic option but uses an order or magnitude more resources as a result.

3.3.3 Custom format

Technically one can use RMAN (or some other piece of software) to create a uncompressed backup of the system. Then one could apply a custom compression format on the backup that is provided with the system. The major advantage with this approach is Unix/Linux typically provide a plethora of free compression formats such as LZ4, BZIP, GZIP, and TAR that can be applied to any sort of data. Using the large list of options, one could hand choose a compression format that suits their needs to get the most optimal space savings and system performance. Additionally if one is storing the backups on a separate system, then they could hypothetically perform the compression on the backup system instead which could free up the main system's resources.

3.4 Discussion

RMAN is a convenient backup/recovery tool that integrates well with a current Oracle based stack. It's basic compression option is available in all systems and achieves considerable space savings with a moderate impact on performance. However Oracle's new RMAN Advanced Compression options can allow you to fine tune the compression based off of your needs unlike the basic option. Also the MEDIUM option that is provided with this tends to get around the same space savings while using less system resources. Then we have using a custom format for compressing your backups, which have no licensing attached and can further fine tune the compression then what RAMN currently offers. However picking an appropriate compression format and fine tuning it will take adequate amount of research and testing to get it right. Additionally using a custom format for compression means that one will have to modify their backup/recovery process since a tool, like RMAN, process the data in a non custom format.

3.5 Conclusion

In conclusion, Oracle currently offers a backup/recovery tool called RMAN that works well with Oracle Databases and has a plethora of features. It has a basic compression option available to all that performs pretty well. Later on Oracle added more advanced compression options with their Advanced Compression license which can allow one to fine tune the compression to their needs (and performs better than the basic option). Then finally we have the custom format that uses the compression format of ones choice on the backups which allows even greater control over space savings and system performance but would require some more work to it pull off.

4 Citations/Links

ZFS

- ZFS, Wikipedia, 18-Nov-2017. [Online]. Available: https://en.wikipedia.org/wiki/ZFS. [Accessed: 21-Nov-2017].
- ZFS Deduplication, ZFS Deduplication, 02-Nov-2009. [Online]. Available: https://blogs.oracle.com/bonwick/zfs-deduplication-v2. [Accessed: 21-Nov-2017].
- 8. Storage, FreeNAS User Guide 9.3. [Online]. Available: https://doc.freenas.org/9.3/freenas_storage.html. [Accessed: 21-Nov-2017].

HCC

- G. Christman and K. Jernigan, Hybrid Columnar Compression (HCC) on Exadata, Nov-2012. [Online]. Available: http://www.oracle.com/technetwork/database/exadata/ehcc-twp-131254.pdf. [Accessed: 21-Nov-2017].
- S. Ledbetter , Implementing Hybrid Columnar Compression on the Oracle ZFS Storage Appliance, May-2014. [Online]. Available: http://www.oracle.com/technetwork/server-storage/sun-unified-storage/documentation/problemsolver-hcc-52014-2202692.pdf. [Accessed: 21-Nov-2017].
- F. Pachot, Oracle compression, availability and licensing, Blog dbi services, 31-Jul-2017. [Online]. Available: https://blog.dbi-services.com/oracle-compression-availability-and-licensing/. [Accessed: 21-Nov-2017].

Index Compression

- Compressing your Indexes: Index Key Compression (PART 1), Compressing your Indexes: Index KOracle Database Storage Optimization Blog, 15-Mar-2016. [Online]. Available: https://blogs.oracle.com/dbstorage/compressing-your-indexes:-index-key-compression-part-1. [Accessed: 21-Nov-2017].
- Compressing your Indexes: Advanced Index Compression (PART 2), Oracle Database Storage Optimization Blog, 25-Mar-2016. [Online]. Available: https://blogs.oracle.com/dbstorage/compressing-your-indexes:-advanced-index-compression-part-2. [Accessed: 21-Nov-2017].

- Advanced Index Compression High - New with Oracle Database 12c Release 2 on Oracle Cloud, Oracle Database Storage Optimization Blog, 06-Jan-2017. [Online]. Available: https://blogs.oracle.com/dbstorage/advanced-index-compression-high-new-with-oracle-database-12c-release-2-on-oracle-cloud. [Accessed: 21-Nov-2017].

Backup Compression

- Oracle Advanced Compression with Oracle Database 12c Release 2, Sep-2017. [Online]. Available: http://www.oracle.com/technetwork/database/options/compression/advanced-compression-wp-12c-1896128.pdf. [Accessed: 21-Nov-2017].
- Database Backup and Recovery User's Guide, Getting Started with RMAN, 13-May-2015. [Online]. Available: https://docs.oracle.com/cd/E11882_01/backup.112/e10642/rcmquick.htm#BRADV89371. [Accessed: 21-Nov-2017].
- R. Blogger, RMAN Compression Algorithms in 11gR2, The Official Rackspace Blog, 01-Mar-2012. [Online]. Available: https://blog.rackspace.com/rman-compression-algorithms-in-11gr2. [Accessed: 21-Nov-2017].
- Comparing of RMAN Backup Compression Levels, Talip Hakan Ozturk's ORACLE BLOG, 07-Apr-2012. [Online]. Available: https://taliphakanozturken.wordpress.com/2012/04/07/comparing-of-rman-backup-compression-levels/. [Accessed: 21-Nov-2017].
- List of archive formats, Wikipedia, 06-Oct-2017. [Online]. Available: https://en.wikipedia.org/wiki/List_of_archive_formats. [Accessed: 21-Nov-2017].