Microsoft SQL Server

Microsoft

SQL Server 2016 In-memory OLTP

Technical White Paper

Writer: Kalen Delaney

Technical Reviewers: Sunil Agarwal, Cristian Diaconu, Craig Freedman, Naveen Prakash, Simon Koeman

Published: May 2015

Applies to: SQL Server 2016 CTP2

Summary: In-memory OLTP, frequently referred to by its codename “Hekaton”, was introduced in SQL Server 2014. This powerful technology allows you to take advantage of large amounts of memory and many dozens of cores to increase performance for OLTP operations by upto 30 to 40 times! SQL Server 2016 is continuing the investment in In-memory OLTP by removing many of the limitations found in SQL Server 2014, and enhancing internal processing algorithms so that In-memory OLTP can provide even greater improvements. This paper gives a high level overview of these changes in SQL Server 2016, focusing on those that are available in CTP2. (Subsequent versions of this paper will discuss features available in CTP3 and well as more technical details describing all the changes, and also further description of the internals of the new processing algorithms.)

Copyright

Introduction

SQL Server was originally designed at a time when it could be assumed that main memory was very expensive, so data needed to reside on disk except when it was actually needed for processing. This assumption is no longer valid as memory prices have dropped enormously over the last 30 years. At the same time, multi-core servers have become affordable, so that today one can buy a server with 32 cores and 1TB of memory for under $50K. Since many, if not most, of the OLTP databases in production can fit entirely in 1TB, we need to re-evaluate the benefit of storing data on disk and incurring the I/O expense when the data needs to be read into memory to be processed. In addition, OLTP databases also incur expenses when this data is updated and needs to be written back out to disk. Memory-optimized tables are stored completely differently than disk-based tables and these new data structures allow the data to be accessed and processed much more efficiently.

Because of this trend to much more available memory and many more cores, the SQL Server team at Microsoft began building a database engine optimized for large main memories and many-core CPUs.

Project Hekaton

Project Hekaton was envisioned as a completely new data storage and access technology that would potentially allow up to a 100-fold increase in performance for OLTP workloads. For the initial In-memory OLTP release in SQL Server 2014, Microsoft customers are already seeing 30- and 40-fold improvements. This increased performance is obtained by making changes in three main areas compared to the previous disk-based storage for our relational data.

1. Data structures

   Completely new data structures have been designed for the rows and indexes of In-memory OLTP. These structures are designed with multi-versioning in mind, and are also designed to be updateable with no locking required.

2. No Locking and Latching

   Locking is one of the most common causes of long wait times and slow responses in a multi-user OLTP system. Because of the way the In-memory OLTP data structures have been designed, no locking is required for any data manipulation operations. In addition, because data is not read in from disk into memory buffers, and the rows are not stored on pages, no page latches are needed for In-memory OLTP data processing.

3. Native compilation

   Even though we talk about ‘recompiling’ for our queries and procedures acceding disk-based tables in SQL Server, the code is not truly compiled. It is translated into a lower level of code that can be interpreted, but it is not compiled into machine code. So execution of normal operations on disk-based tables requires line by line interpretation of each operation. But In-memory OLTP supports natively compiled procedures that can access memory-optimized tables that will be truly compiled and then loaded as DLLs.

These major changes in data storage and access, plus many smaller changes, including much more efficient log writes, allow the incredible performance improvements obtained with SQL Server 2014 In-memory OLTP.

Changes in SQL Server 2016

This paper is written for the CTP2 release of SQL Server 2016 and intended as an overview of the changes. This paper will be updated for the CTP3 release to include a more detailed look at SQL Server 2016 In-memory OLTP. A final version of this whitepaper will be available near the time of the RTM release of the SQL Server 2016 product.

In the CTP2 release of SQL Server 2016, you will find many enhancements to the technology, and many limitations removed. Many of these changes are described below.

Maximum memory for memory-optimized tables

In SQL Server 2014, there is a recommendation that you don't use more than 256 GB of memory for memory-optimized tables. This is not a hard limit, just a recommendation, based on the number of checkpoint files that can be supported to store the data from your memory-optimized tables, which are used during the recovery process. In SQL Server 2016, there is no hard limit on the number of checkpoint files. However, Microsoft only tests up to a certain size. As you have more memory used for memory-optimized tables, there is more overhead in related areas, so there is still a recommended maximum. In SQL Server 2016, the recommendation is not to have more than 2 TB of data in memory-optimized tables. Of course, no matter how much data you have in your memory-optimized tables, all the data in all the memory-optimized tables must fit in memory.

Collation

In-memory OLTP in SQL Server 2014 requires that any character columns that are used as all or part of an index key must use a BIN2 collation. In addition, in a natively compiled procedure, all comparisons between character values must be a BIN2 collation. The collation can be defined as a database default, so that all character columns use it, or it can be specified as an attribute of the column when the table is recreated. In SQL Server 2016, character columns using any collation can be part of an index, and comparisons between character values using any collations are supported. Be aware, though, that there can be a measurable performance penalty when using collations other than BIN2.

Schema and data changes

SQL Server 2014 does not allow any changes to memory-optimized tables after creation. In SQL Server 2016, ALTER TABLE can be used on memory-optimized tables to add, drop or alter columns, or to add, drop or rebuild indexes.

Parallel plans

SQL Server 2014's optimizer will never create a parallel plan for operations accessing memory-optimized tables. In SQL Server 2016, certain operations that use hash indexes can be performed in parallel, if they're not used in a natively compiled procedure.

Transparent Data Encryption

TDE will be supported in SQL Server 2016. In SQL Server 2014, a database with encryption enabled would not encrypt data stored on a MEMORY_OPTIMIZED_DATA filegroup. That limitation goes away in SQL Server 2016 and memory-optimized tables data on disk will be encrypted.

Native compilation

SQL Server 2014 limited the language constructs available in natively compiled procedures to the simplest DML operations needed for basic transaction processing. In SQL Server 2016 CTP2, natively compiled procedures will support a wider range of features, including the following constructs:

- LEFT and RIGHT OUTER JOIN
- SELECT DISTINCT
- OR and NOT operators
- Subqueries in all clauses of a SELECT statement
- Nested stored procedure calls
- UNION and UNION ALL
- All built-in math functions

Changes to processing algorithms

In addition to expanding the features supported, SQL Server 2016 CTP2 is changing some of the internal processing mechanisms used for In-memory OLTP.

Dependency on Windows filestream processing

SQL Server 2014 used the underlying Windows operating system to manage writes to the data files, using its filestream technology to allocate and manage the files. However, filestream requires a SQL Server table (*xtp_storage*) that contains a filestream column backed up by the MEMORY_OPTIMIZED_DATA filegroup.

In SQL Server 2016, filestream is now used only to provide a user-visible surface for the definition of the MEMORY_OPTIMIZED_DATA filegroup. File management for checkpoint files for In-memory OLTP has been separated from filestream.The In-memory OLTP engine allocates, resizes and deletes the files itself by calling NTFS APIs directly. At the user level, you'll be able to see changes because the directory structure of the filegroup is different in SQL Server 2016.

A side-effect of this change is that file collection no longer depends on the filestream garbage collection. The only files you should be able to see on disk in the In-memory OLTP containers should be files that are still referenced by the database log.

Multiple log reader threads

SQL Server 2014 uses only one log reader thread [per database] to read transactions affecting memory-optimized tables from the SQL Server transaction log. This single thread reads the tail of the log to load into memory during recovery, it moves rows from the log to the data and delta files, and it applies log stream modifications to the in-memory data rows when the system is using AlwaysOn. Having one thread for all log reading operations created a scalability bottleneck and a ceiling to the speed of these operations.

SQL Server 2016 will allow multiple threads for both recovery and checkpoint, to read and apply the logged transactions, allowing much greater scalability for very high volume OLTP applications. These processes will basically be scalable with the number of cores.

Number of sockets

SQL Server 2014 had limited scalability with multiple socket machines. In SQL Server 2016, you can expect efficient scaling with a 4-socket machine. This is not due to particular features being changed or added, but just to improvements in the existing algorithms. This includes better partitioning of worktime queues and more capacity in the threads assigned to do garbage collection of the in-memory data.

AlwaysOn

In SQL Server 2014, the data visibility of in-memory OLTP on the secondary replica was delayed by few transactions. This limitation is now removed. The goal in SQL Server 2016 is for all data, from both disk-based tables and memory-optimized tables, to be visible to a human user at the same time. You should not be able to commit a transaction on the primary and then, upon checking the state on the secondary, miss the modification.

Garbage Collection in memory

SQL Server 2014 In-memory OLTP uses internal garbage collection threads to clean up row versions that are no longer needed. In extremely high-volume workloads, it is sometimes the case that the garbage collection cannot keep up with the volume of versions needed to be removed. One area in which this is frequently noticeable is when a large memory-optimized table is dropped. There can be a noticeable delay before the memory used by the dropped table is available for new rows.

In SQL Server 2016, not only DROP of a table, but the new ability to ALTER a table can generate large numbers of row versions. The garbage collection algorithms have been improved to provide greater scalability. With sufficient processing power, the garbage collection for your DML operations will happen almost instantaneously.

For a description and usage information for SQL Server 2014 In-memory OLTP, please refer to the Books Online documentation: http://go.microsoft.com/fwlink/p/?LinkId=536616

For more internal details of the SQL Server 2014 In-memory OLTP implementation, take a look at the whitepaper written for the CTP2 release: SQL_Server_2014_In-Memory_OLTP_TDM_White_Paper.pdf

Additional details, and descriptions of changes for the SQL Server 2014 RTM release, were incorporated into a book published by Red Gate: http://www.red-gate.com/community/books/sql-server-internals-in-memory-oltp

Conclusion

SQL Server In-memory OLTP is a new data processing technology that allows a many-fold improvement in performance for OLTP applications. The biggest improvements are due to completely new storage mechanisms, complete optimistic concurrency with no locking or latching for any operations and native compilation of stored procedures. In-memory OLTP was introduced in SQL Server 2014 and has been enhanced for SQL Server 2016. This paper describes the new features added and the changes in some of the internal processing that was introduced in SQL Server 2016 CTP2.

For more information:

http://www.microsoft.com/sqlserver/: SQL Server Web site

http://technet.microsoft.com/en-us/sqlserver/: SQL Server TechCenter

http://msdn.microsoft.com/en-us/sqlserver/: SQL Server DevCenter

Did this paper help you? Please give us your feedback. Tell us on a scale of 1 (poor) to 5 (excellent), how would you rate this paper and why have you given it this rating? For example:

- Are you rating it high due to having good examples, excellent screen shots, clear writing, or another reason?
- Are you rating it low due to poor examples, fuzzy screen shots, or unclear writing?

This feedback will help us improve the quality of white papers we release.

Send feedback.