

SQL Server 2014 Columnstore Index Improvements

Bob Beauchemin
bobb@SQLskills.com



Introduction

- **Columnstores**
 - Column-based storage vs. Row-based storage
 - Clustered columnstore
 - Archival compression
- **Query Improvements**
 - Batch mode
 - Partition and segment elimination
- **Updatable Columnstore**
 - Deltastores
 - Insert batch improvement
- **Use cases**
 - For clustered and nonclustered columnstore



OLAP and OLTP

- **OLAP and OLTP are general categories for data applications**
 - OLTP applications
 - Read one or very few rows at a time
 - Read all or many of the columns in a row
 - OLAP applications
 - Reads many rows – scans or range scans
 - Reads very few columns from a row
- **Columnstores are a good choice for OLAP applications**
- **Columnstores are significantly slower for OLTP applications**
- **Columnstore is an Enterprise feature**

Columnstore and Data Warehouse

- **Data warehouse is the primary use case**
- **Data warehouse star or snowflake schema consists of**
 - Fact Tables – large number of rows, wide rows
 - Dimension Tables – Small tables to group by
- **Fact Tables are notoriously difficult to index**
 - Aggregates are pre-calculated offline for speed
 - Columnstores help here
 - May also help with large dimension tables

xVelocity

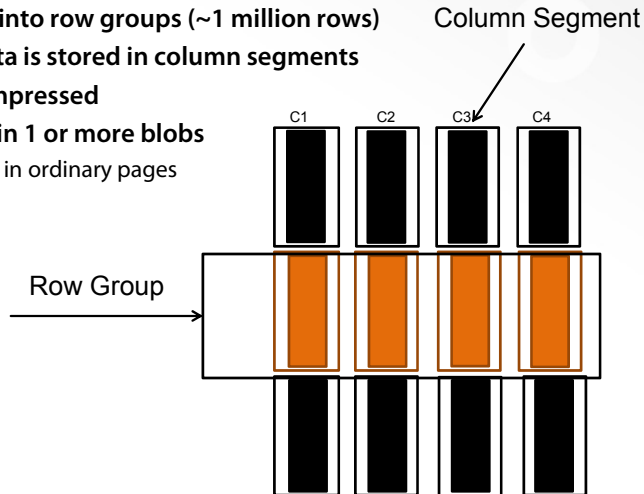
- **Based on xVelocity technology**
 - Introduced in PowerPivot (was Vertipac)
 - In SQL Server 2012, **implemented in database engine** and SSAS
- **Profound speedup for data warehouse queries**
 - 10-100 times faster
 - Not all queries benefit

Column-Based Storage

- **SQL Server data is customarily stored in rows**
 - Rows are stored in pages
 - Pages hold 1-n rows
- **Columnstore data is stored on a per-column basis**
 - Column segments contain values from the same column
 - Segment is the unit of transfer between disk and memory
 - Rowgroups refer to sets of column segments from columns in same table
 - Technically, column segments are stored as blobs
 - And data is still stored in pages

Row Groups and Column Segments

- Rows are divided into row groups (~1 million rows)
- Each column's data is stored in column segments
- Segments are compressed
- Segments stored in 1 or more blobs
 - Blobs are stored in ordinary pages



Columnstore Indexes

- Columnstores are known as "columnstore indexes" (CCI)
- If columnstore is the only storage,
 - This is "clustered columnstore index"
 - Not related conceptually to clustered B-tree index
 - Clustered columnstore index new in SQL Server 2014
- If both row store and a columnstore exist (NCCI)
 - This is "nonclustered columnstore index"
 - Row store can contain columns not in columnstore
 - Introduced in SQL Server 2012
- Either type of index can be partitioned

Building Columnstores

- **To build clustered columnstore index**
 - Create heap or table
 - Or use existing table
 - Drop nonclustered indexes (if any) and constraints
 - All column data types must be supported by columnstore index
 - CREATE CLUSTERED COLUMNSTORE INDEX
 - ...WITH DROP_EXISTING = TRUE
 - Uses all columns in table, all column types must be supported
- **To build nonclustered columnstore index**
 - CREATE COLUMNSTORE INDEX
 - With all columns or subset of columns
 - One columnstore index/table
 - Must be partition-aligned if used with partitioning
 - Makes table read-only

Data Type Support

- **Unsupported Types**
 - Decimal with precision greater than 18
 - Binary
 - BLOB
 - (n)varchar(max)
 - Uniqueidentifier
 - Datetimeoffset with precision greater than 2
 - CLR (includes spatial)
- **SQL Server 2014 includes**
 - All decimal
 - All datetimeoffset
 - Uniqueidentifier
 - Binary,varbinary

Compression

- **Various compression techniques used**
 - Dictionary Encoding
 - Run-length Encoding
 - Huffman Encoding
 - Lempel-Ziv-Welch
 - Value Encoding
- **Archival compression in SQL Server 2014**
 - Adds another layer of compression using XPress8 (a variant of LZ77)
- **Dictionary used for:**
 - All string columns
 - Non-string columns with few distinct values
- **Two types of dictionary**
 - Primary: One per column
 - Created only during index build
 - Secondary (overflow): 0-nper column
 - Each segment has 0 or 1 secondary dictionary
 - Secondary dictionary may be used by more segments

Metadata Tables

- **sys.column_store_segments**
 - Shows one row per segment / per index partition
- **sys.column_store_dictionaries**
 - If dictionaries used for columnstore compression
- **sys.column_store_row_groups (SQL Server 2014 only)**
 - Can distinguish between deltastore and segment
 - Shows status and status description

Performance Improvements

- **Storage format**
 - Columnstores reduce IO
 - Smart compression and encoding reduce memory and disk utilization
 - Improved buffer pool usage
- **Query Improvements**
 - Uses less memory during queries
 - Batch mode query iterators

SQL Server 2014 Changes

- **Columnstore can be the only storage for a table**
 - Known as clustered columnstore index
 - If so, that table is updateable
 - Updates use deltastore (more later)
 - Schema can be changed
- **For clustered and nonclustered columnstore**
 - Additional data compression type
 - Columnstore archive compression
 - More data types are supported
 - Improved global dictionaries for better compression
 - Short string support without using dictionaries

Columnstore and Queries

- **Optimizer is still cost-based**
 - Optimizer can choose index
 - Optimizer can choose batch mode
 - Batch mode not always used
- **Can observe in query plan**
 - Columnstore index use
 - Batch execution mode
- **Columnstore queries can benefit from**
 - Segment elimination
 - Partition elimination (if partitioning used)
- **Some queries slower with columnstore**

Batch Mode Execution

- **Query processor usually uses row mode**
 - Query operators process one row at a time
- **SQL Server 2012 added batch mode**
 - Also known as "vectorized execution"
 - Operators handle ~1000 rows at a time
 - Saves CPU resources - less instructions executed
- **Batch mode can be observed in**
 - Query plan (on a per-operator basis)
 - Query execution speed

SQL Statement Improvements - SQL 2014

- More iterators support batch mode
- Mixed-mode plans (batch and row) supported
- Seek operation support
- Bulk inserts optimized with clustered columnstore indexes

Batch Mode Restrictions – SQL Server 2012

- Restrictions (in SQL Server 2012)
 - Batch Processing iterators not used for
 - Outer joins
 - Union
 - Estimated/Actual execution mode = Row
 - NOT batch
 - May be able to rewrite query to use Batch Processing

Query Hints (Nonclustered Columnstore)

- **Nonclustered Columnstore-specific query hints**
 - USE columnstore index (traditional index hint)
 - USE non-columnstore index (traditional index hint)
 - OPTION(ignore_nonclustered_columnstore_index)

SQL Server 2014 Batch Mode Improvements

- **Batch Mode Hash Join Support**
 - Improved batch-mode hash join handles inner, outer, semi and anti-semi joins
 - Batch mode hash joins handle spilling better
- **Union all supported**
 - But not union
- **Partial aggregation**
 - Global aggregation use row mode
 - Distinct aggregates use row mode

Updating with Nonclustered Columnstore

- **Table cannot be updated directly**
- **During nightly batch processing**
 - Disable (or drop) columnstore index
 - Nightly load (update table)
 - Rebuild columnstore index
- **Table can also be "updated" via partition switching**
 - Load new data into staging table
 - Build columnstore index
 - Split empty partition
 - Switch the staging table partition into table
- **Variations possible for more frequent updates**
 - Trickle loading

Updating Clustered Columnstore

- **Updates use 1-n deltastores**
 - This is a row store, not a columnstore
 - Insert – to deltastore
 - Delete – logical operation, data removed on next rebuild
 - Update – Insert and Delete pair
 - Bulk Insert – Goes to the columnstore if large enough (> 100k)
 - Otherwise, goes to delta store
 - Select – uses deltastore and columnstore (union internally)

Deltastore Behavior

- **Disadvantages of Deltastore**
 - The entire row has to be read
 - Segment elimination does not occur for deltastores
 - Parallel scans will distribute the deltastores among the threads so that multiple deltastores are scanned in parallel
 - No inter-deltastore parallelism
 - Concurrent deletes or updates **are blocked** while the Tuple Mover compresses a deltastore.
 - Concurrent Inserts are not blocked by the Tuple Mover.
- **Delete bitmap**
 - Only compressed segments use a deleted bitmap

Updating and Columnstore Segments

- **Conversion of rows (delta store) to columnstore uses tuple-mover**
 - Automatic when segment is full
 - Index reorganize starts it manually
 - Manual invoke results in undersized segments
- **Bulk Loading is better way to update**
 - Batches of less than ~100000 rows loaded as segments
 - Batches of less than ~100000 rows loaded into deltastore
- **Rebuilding index more useful**
 - Rebuilding index can make optimum size segments
 - Rebuilding index rebuilds the global dictionary
 - Reorganize index doesn't

Better Memory Management – SQL Server 2014

- **When building columnstore**
 - Adjusts to low memory condition
- **Runtime**
 - Batch mode spilling not longer reverts to row mode
 - Presence of row operators doesn't prohibit batch mode
- **Honors resource governor**

Use Cases and Best Practices

- **Usually good for fact tables in data warehouse**
 - Maybe large dimension tables
- **Fits with data warehouse pattern of**
 - Read-only tables used for analysis
 - Nightly job to append data
- **Queries aggregate lots of base rows**
- **Columnstore is relational table-based**
 - Can use PowerPivot
 - Can use SSAS instead (in SQL Server 2012)
- **Choose your xVelocity vehicle**

When to use Nonclustered (SQL Server 2014)

- **You must use nonclustered columnstore**
 - If you need key constraints or triggers
 - If you have unsupported data types
 - Consider refactoring
 - Consider refactoring for string types in any case
- **You should use clustered columnstore**
 - One copy of the data (you won't need other indexes)

Review

- **Columnstore improvements**
 - Clustered columnstore
 - For nonclustered columnstore too
- **Archival compression**
- **Batch mode improvements**
 - For clustered and nonclustered queries
- **Deltastores and updating clustered columnstore**
 - Optimizing batch inserts
- **Memory management improvements**

References

- **SQL Server 2014 Books Online**
- **Columnstore Index Articles on Technet Wiki**
 - SQL Server Columnstore Index FAQ
 - <http://bit.ly/HMfRKw>
 - SQL Server Columnstore Performance Tuning
 - <http://bit.ly/wt0CLj>
 - Others...
- **Blogs**
 - Remus Rusanu
 - <http://rusanu.com/blog/>
 - Niko Neugebauer (series on clustered columnstore)
 - <http://bit.ly/1d8ykxB>