

Columnstore Best Practices

Last updated by | Vitor Tomaz | Feb 24, 2023 at 3:28 AM PST

Contents

- [Typical usage scenarios for columnstore indexes](#)
- [Design guidance](#)
- [Query performance](#)
- [Index maintenance](#)
- [Public Doc Reference](#)

Columnstore Index Best Practices

This is a "How To" TSG that provides information about performance best practices for columnstore indexes.

For a general introduction to columnstore indexes, see article [Columnstore indexes: Overview](#) .









Typical usage scenarios for columnstore indexes

Columnstore indexes are typically used for storing and querying large data warehousing fact tables. They are also suitable for analytic-type queries that scan large ranges of values. Columnstore indexes are designed to work well for large range scans rather than looking up specific values. Columnstore indexes work well when the data is stable, e.g. when queries update and delete less than 10% of the rows. But columnstore indexes might also be feasible for transactional workloads, e.g. by adding a nonclustered columnstore index on a rowstore storage-based table or a rowstore in-memory OLTP table.

Although columnstore indexes can increase the performance for specific scenarios, if it is not designed or maintained properly, it can generate severe performance issues.












Design guidance

This topic is discussed in depth in the public article [Columnstore indexes - Design guidance](#) . The following topics are included:




- [Choose the best columnstore index for your needs](#) 
- [Use a clustered columnstore index for large data warehouse tables](#) 
- [Use an ordered clustered columnstore index for large data warehouse tables](#) 
- [Add B-tree nonclustered indexes for efficient table seeks](#) 
- [Use a nonclustered columnstore index for real-time analytics](#) 
- [Use table partitions for data management and query performance](#) 
- [Choose the appropriate data compression method](#) 
- [Use optimizations when you convert a rowstore table to a columnstore index](#) 



Query performance


This topic is discussed in depth in the public article [Columnstore indexes - Query performance](#) . The following topics are included:

- [Recommendations for improving query performance](#) 
 - [1. Organize data to eliminate more rowgroups from a full table scan](#) 
 - [2. Plan for enough memory to create columnstore indexes in parallel](#) 
- [Columnstore Performance Explained](#) 
 - [Data compression](#) 
 - [Column elimination](#) 
 - [Rowgroup elimination](#) 
 - [Batch Mode Execution](#) 
 - [Aggregate pushdown](#) 
 - [String predicate pushdown](#) 
- [Segment elimination](#) 




Index maintenance

This topic is covered in depth in the public article [Optimize index maintenance to improve query performance and reduce resource consumption](#)  and discusses rowstore vs. columnstore topics. It helps you decide when and how to perform index maintenance. It covers concepts such as index fragmentation and page density, and their impact on query performance and resource consumption. It describes index maintenance methods, [reorganizing an index](#)  and [rebuilding an index](#) , and suggests an index maintenance strategy that balances potential performance improvements against resource consumption required for maintenance.

Also note the [Considerations specific to rebuilding a columnstore index](#)  and [Considerations specific to reorganizing a columnstore index](#)  with their counter-intuitive conclusion that a reorg is actually better than a rebuild in most scenarios.

And don't miss the the section [Index maintenance in Azure SQL Database and Azure SQL Managed Instance](#)  towards the bottom of that article for a more general considerations on this topic.

Public Doc Reference

- [Columnstore index design guidelines](#)  - *explains internals and architecture concisely and step-by-step*
- [Get started with Columnstore for real-time operational analytics](#) 
- [Clustered Columnstore Index in Azure SQL Database](#)  - *older article with a short example script*

How good have you found this content?

