**Parameters**

**Databases**

Select databases. The keywords SYSTEM\_DATABASES, USER\_DATABASES, ALL\_DATABASES, and AVAILABILITY\_GROUP\_DATABASES are supported. The hyphen character (-) is used to exclude databases, and the percent character (%) is used for wildcard selection. All of these operations can be combined by using the comma (,).

|  |  |
| --- | --- |
| **Value** | **Description** |
| SYSTEM\_DATABASES | All system databases (master, msdb, and model) |
| USER\_DATABASES | All user databases |
| ALL\_DATABASES | All databases |
| AVAILABILITY\_GROUP\_DATABASES | All databases in availability groups |
| USER\_DATABASES, -AVAILABILITY\_GROUP\_DATABASES | All user databases that are not in availability groups |
| Db1 | The database Db1 |
| Db1, Db2 | The databases Db1 and Db2 |
| USER\_DATABASES, -Db1 | All user databases, except Db1 |
| %Db% | All databases that have “Db” in the name |
| %Db%, -Db1 | All databases that have “Db” in the name, except Db1 |
| ALL\_DATABASES, -%Db% | All databases that do not have “Db” in the name |

**FragmentationLow**

Specify index maintenance operations to be performed on a low-fragmented index.

|  |  |
| --- | --- |
| **Value** | **Description** |
| INDEX\_REBUILD\_ONLINE | Rebuild index online. |
| INDEX\_REBUILD\_OFFLINE | Rebuild index offline. |
| INDEX\_REORGANIZE | Reorganize index. |
| INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE | Rebuild index online. Rebuild index offline if online rebuilding is not supported on an index. |
| INDEX\_REBUILD\_ONLINE,INDEX\_REORGANIZE | Rebuild index online. Reorganize index if online rebuilding is not supported on an index. |
| INDEX\_REORGANIZE,INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE | Reorganize index. Rebuild index online if reorganizing is not supported on an index. Rebuild index offline if reorganizing and online rebuilding are not supported on an index. |
| NULL | Do not perform index maintenance. This is the default for a low-fragmented index. |

An online index rebuild or an index reorganization is not always possible. Because of this, you can specify multiple index-maintenance operations for each fragmentation group. These operations are prioritized from left to right: If the first operation is supported for the index, then that operation is used; if the first operation is not supported, then the second operation is used (if supported), and so on. If none of the specified operations are supported for an index, then that index is not maintained.

IndexOptimize uses the SQL Server [ALTER INDEX](https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-index-transact-sql) command: REBUILD WITH (ONLINE = ON) to rebuild indexes online, REBUILD WITH (ONLINE = OFF) to rebuild indexes offline, and REORGANIZE to reorganize indexes.

**FragmentationMedium**

Specify index maintenance operations to be performed on a medium-fragmented index.

|  |  |
| --- | --- |
| **Value** | **Description** |
| INDEX\_REBUILD\_ONLINE | Rebuild index online. |
| INDEX\_REBUILD\_OFFLINE | Rebuild index offline. |
| INDEX\_REORGANIZE | Reorganize index. |
| INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE | Rebuild index online. Rebuild index offline if online rebuilding is not supported on an index. |
| INDEX\_REBUILD\_ONLINE,INDEX\_REORGANIZE | Rebuild index online. Reorganize index if online rebuilding is not supported on an index. |
| INDEX\_REORGANIZE,INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE | Reorganize index. Rebuild index online if reorganizing is not supported on an index. Rebuild index offline if reorganizing and online rebuilding are not supported on an index. This is the default for a medium-fragmented index. |
| NULL | Do not perform index maintenance. |

An online index rebuild or an index reorganization is not always possible. Because of this, you can specify multiple index-maintenance operations for each fragmentation group. These operations are prioritized from left to right: If the first operation is supported for the index, then that operation is used; if the first operation is not supported, then the second operation is used (if supported), and so on. If none of the specified operations are supported for an index, then that index is not maintained.

IndexOptimize uses the SQL Server [ALTER INDEX](https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-index-transact-sql) command: REBUILD WITH (ONLINE = ON) to rebuild indexes online, REBUILD WITH (ONLINE = OFF) to rebuild indexes offline, and REORGANIZE to reorganize indexes.

**FragmentationHigh**

Specify index maintenance operations to be performed on a high-fragmented index.

|  |  |
| --- | --- |
| **Value** | **Description** |
| INDEX\_REBUILD\_ONLINE | Rebuild index online. |
| INDEX\_REBUILD\_OFFLINE | Rebuild index offline. |
| INDEX\_REORGANIZE | Reorganize index. |
| INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE | Rebuild index online. Rebuild index offline if online rebuilding is not supported on an index. This is the default for a high-fragmented index. |
| INDEX\_REBUILD\_ONLINE,INDEX\_REORGANIZE | Rebuild index online. Reorganize index if online rebuilding is not supported on an index. |
| INDEX\_REORGANIZE,INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE | Reorganize index. Rebuild index online if reorganizing is not supported on an index. Rebuild index offline if reorganizing and online rebuilding are not supported on an index. |
| NULL | Do not perform index maintenance. |

An online index rebuild or an index reorganization is not always possible. Because of this, you can specify multiple index-maintenance operations for each fragmentation group. These operations are prioritized from left to right: If the first operation is supported for the index, then that operation is used; if the first operation is not supported, then the second operation is used (if supported), and so on. If none of the specified operations are supported for an index, then that index is not maintained.

IndexOptimize uses the SQL Server [ALTER INDEX](https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-index-transact-sql) command: REBUILD WITH (ONLINE = ON) to rebuild indexes online, REBUILD WITH (ONLINE = OFF) to rebuild indexes offline, and REORGANIZE to reorganize indexes.

**FragmentationLevel1**

Set the lower limit, as a percentage, for medium fragmentation. The default is 5 percent. This is based on Microsoft’s recommendation in [Books Online](https://docs.microsoft.com/en-us/sql/relational-databases/indexes/reorganize-and-rebuild-indexes).

IndexOptimize checks avg\_fragmentation\_in\_percent in [sys.dm\_db\_index\_physical\_stats](https://docs.microsoft.com/en-us/sql/relational-databases/system-dynamic-management-views/sys-dm-db-index-physical-stats-transact-sql) to determine the fragmentation.

**FragmentationLevel2**

Set the lower limit, as a percentage, for high fragmentation. The default is 30 percent. This is based on Microsoft’s recommendation in [Books Online](https://docs.microsoft.com/en-us/sql/relational-databases/indexes/reorganize-and-rebuild-indexes).

IndexOptimize checks avg\_fragmentation\_in\_percent in [sys.dm\_db\_index\_physical\_stats](https://docs.microsoft.com/en-us/sql/relational-databases/system-dynamic-management-views/sys-dm-db-index-physical-stats-transact-sql) to determine the fragmentation.

**PageCountLevel**

Set a size, in pages; indexes with fewer pages are skipped for index maintenance. The default is 1000 pages. This is based on Microsoft’s recommendation.

IndexOptimize checks page\_count in [sys.dm\_db\_index\_physical\_stats](https://docs.microsoft.com/en-us/sql/relational-databases/system-dynamic-management-views/sys-dm-db-index-physical-stats-transact-sql) to determine the size of the index.

**SortInTempdb**

Use tempdb for sort operations when rebuilding indexes.

|  |  |
| --- | --- |
| **Value** | **Description** |
| Y | Use tempdb for sort operations when rebuilding indexes. |
| N | Do not use tempdb for sort operations when rebuilding indexes. This is the default. |

The SortInTempdb option in IndexOptimize uses the SORT\_IN\_TEMPDB option in the SQL Server [ALTER INDEX](https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-index-transact-sql) command.

**MaxDOP**

Specify the number of CPUs to use when rebuilding indexes. If this number is not specified, the global maximum degree of parallelism is used.

The MaxDOP option in IndexOptimize uses the MAXDOP option in the SQL Server [ALTER INDEX](https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-index-transact-sql) command.

**FillFactor**

Indicate, as a percentage, how full the pages should be made when rebuilding indexes. If a percentage is not specified, the fill factor in [sys.indexes](https://docs.microsoft.com/en-us/sql/relational-databases/system-catalog-views/sys-indexes-transact-sql) is used.

The FillFactor option in IndexOptimize uses the FILLFACTOR option in the SQL Server [ALTER INDEX](https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-index-transact-sql) command.

**PadIndex**

Apply the percentage of free space that the fill factor specifies to the intermediate-level pages of the index.

|  |  |
| --- | --- |
| **Value** | **Description** |
| Y | Apply the percentage of free space that the fill factor specifies to the intermediate-level pages of the index. |
| N | The intermediate-level pages of the index are filled to near capacity. This is the default. |

The PadIndex option in IndexOptimize uses the PADINDEX option in the SQL Server [ALTER INDEX](https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-index-transact-sql) command.

**LOBCompaction**

Compact pages that contain large object (LOB) columns, when reorganizing indexes.

|  |  |
| --- | --- |
| **Value** | **Description** |
| Y | Compact pages that contain LOB columns, when reorganizing indexes. This is the default. |
| N | Do not compact pages that contain LOB columns, when reorganizing indexes. |

The LOBCompaction option in IndexOptimize uses the LOB\_COMPACTION option in the SQL Server [ALTER INDEX](https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-index-transact-sql) command.

**UpdateStatistics**

Update statistics.

|  |  |
| --- | --- |
| **Value** | **Description** |
| ALL | Update index and column statistics. |
| INDEX | Update index statistics. |
| COLUMNS | Update column statistics. |
| NULL | Do not perform statistics maintenance. This is the default. |

IndexOptimize uses the SQL Server [UPDATE STATISTICS](https://docs.microsoft.com/en-us/sql/t-sql/statements/update-statistics-transact-sql) command to update statistics.

**OnlyModifiedStatistics**

Update statistics only if any rows have been modified since the most recent statistics update.

|  |  |
| --- | --- |
| **Value** | **Description** |
| Y | Update statistics only if any rows have been modified since the most recent statistics update. |
| N | Update statistics regardless of whether any rows have been modified. |

IndexOptimize checks modification\_counter in [sys.dm\_db\_stats\_properties](https://docs.microsoft.com/en-us/sql/relational-databases/system-dynamic-management-views/sys-dm-db-stats-properties-transact-sql), in SQL Server 2008 R2 starting with Service Pack 2 and in SQL Server 2012 starting with Service Pack 1, to determine whether any rows have been modified since the most recent statistics update. In earlier versions it checks rowmodctr in [sys.sysindexes](https://docs.microsoft.com/en-us/sql/relational-databases/system-compatibility-views/sys-sysindexes-transact-sql).

**StatisticsSample**

Indicate, as a percentage, how much of a table is gathered when updating statistics. A value of 100 is equivalent to a full scan. If no value is specified, then SQL Server automatically computes the required sample.

The StatisticsSample option in IndexOptimize uses the SAMPLE and FULLSCAN options in the SQL Server [UPDATE STATISTICS](https://docs.microsoft.com/en-us/sql/t-sql/statements/update-statistics-transact-sql) command.

**StatisticsResample**

Update statistics with the most recent sample.

|  |  |
| --- | --- |
| **Value** | **Description** |
| Y | Update statistics with the most recent sample. |
| N | Let SQL Server automatically compute the required sample. This is the default. |

The StatisticsResample option in IndexOptimize uses the RESAMPLE option in the SQL Server [UPDATE STATISTICS](https://docs.microsoft.com/en-us/sql/t-sql/statements/update-statistics-transact-sql) command.

You cannot combine the options StatisticsSample and StatisticsResample.

**PartitionLevel**

Maintain partitioned indexes on the partition level. If this parameter is set to Y, the fragmentation level and page count is checked for each partition. The appropriate index maintenance (reorganize or rebuild) is then performed for each partition.

|  |  |
| --- | --- |
| **Value** | **Description** |
| Y | Maintain partitioned indexes on the partition level. This is the default. |
| N | Maintain partitioned indexes on the index level. |

**MSShippedObjects**

Maintain indexes and statistics on objects that are created by internal SQL Server components.

|  |  |
| --- | --- |
| **Value** | **Description** |
| Y | Maintain indexes and statistics on objects that are created by internal SQL Server components. |
| N | Do not maintain indexes and statistics on objects that are created by internal SQL Server components. This is the default. |

IndexOptimize checks is\_ms\_shipped in [sys.objects](https://docs.microsoft.com/en-us/sql/relational-databases/system-catalog-views/sys-objects-transact-sql) to determine whether an object was created by an internal SQL Server component.

**Indexes**

Select indexes. If this parameter is not specified, all indexes are selected. The ALL\_INDEXES keyword is supported. The hyphen character (-) is used to exclude indexes, and the percent character (%) is used for wildcard selection. All these operations can be combined by using the comma (,).

|  |  |
| --- | --- |
| **Value** | **Description** |
| ALL\_INDEXES | All indexes |
| Db1.Schema1.Tbl1.Idx1 | The index Idx1 on the object Schema1.Tbl1 in the database Db1 |
| Db1.Schema1.Tbl1.Idx1, Db2.Schema2.Tbl2.Idx2 | The index Idx1 on the object Schema1.Tbl1 in the database Db1 and the index Idx2 on the object Schema2.Tbl2 in the database Db2 |
| Db1.Schema1.Tbl1 | All indexes on the object Schema1.Tbl1 in the database Db1 |
| Db1.Schema1.Tbl1, Db2.Schema2.Tbl2 | All indexes on the object Schema1.Tbl1 in the database Db1 and all indexes on the object Schema2.Tbl2 in the database Db2 |
| Db1.Schema1.% | All indexes in the schema Schema1 in the database Db1 |
| %.Schema1.% | All indexes in the schema Schema1 in all databases |
| ALL\_INDEXES, -Db1.Schema1.Tbl1.Idx1 | All indexes except the index Idx1 on the object Schema1.Tbl1 in the database Db1 |
| ALL\_INDEXES, -Db1.Schema1.Tbl1 | All indexes except indexes on the object Schema1.Tbl1 in the database Db1 |

**TimeLimit**

Set the time, in seconds, after which no commands are executed. By default, the time is not limited.

**Delay**

Set the delay, in seconds, between index commands. By default, there is no delay.

**WaitAtLowPriorityMaxDuration**

The time, in minutes that an online index rebuild operation will wait for low priority locks.

The WaitAtLowPriorityMaxDuration option in IndexOptimize uses the WAIT\_AT\_LOW\_PRIORITY and MAX\_DURATION options in the SQL Server [ALTER INDEX](https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-index-transact-sql) command.

**WaitAtLowPriorityAbortAfterWait**

The action that will be performed after an online index rebuild operation has been waiting for low priority locks.

|  |  |
| --- | --- |
| **Value** | **Description** |
| NONE | Continue waiting for locks with normal priority. |
| SELF | Abort the online index rebuild operation. |
| BLOCKERS | Kill user transactions that block the online index rebuild operation. |

The WaitAtLowPriorityAbortAfterWait option in IndexOptimize uses the WAIT\_AT\_LOW\_PRIORITY and ABORT\_AFTER\_WAIT options in the SQL Server [ALTER INDEX](https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-index-transact-sql) command.

**AvailabilityGroups**

Select availability groups. The keyword ALL\_AVAILABILITY\_GROUPS is supported. The hyphen character (-) is used to exclude availability groups, and the percent character (%) is used for wildcard selection. All of these operations can be combined by using the comma (,).

|  |  |
| --- | --- |
| **Value** | **Description** |
| ALL\_AVAILABILITY\_GROUPS | All availability groups |
| AG1 | The availability group AG1 |
| AG1, AG2 | The availability groups AG1 and AG1 |
| ALL\_AVAILABILITY\_GROUPS, -AG1 | All availability groups, except AG1 |
| %AG% | All availability groups that have “AG” in the name |
| %AG%, -AG1 | All availability groups that have “AG” in the name, except AG1 |
| ALL\_AVAILABILITY\_GROUPS, -%AG% | All availability groups that do not have “AG” in the name |

**LockTimeout**

Set the time, in seconds, that a command waits for a lock to be released. By default, the time is not limited.

The LockTimeout option in IndexOptimize uses the [SET LOCK\_TIMEOUT](https://docs.microsoft.com/en-us/sql/t-sql/statements/set-lock-timeout-transact-sql) set statement in SQL Server.

**LogToTable**

Log commands to the table dbo.CommandLog.

|  |  |
| --- | --- |
| **Value** | **Description** |
| Y | Log commands to the table. |
| N | Do not log commands to the table. This is the default. |

**Execute**

Execute commands. By default, the commands are executed normally. If this parameter is set to N, then the commands are printed only.

|  |  |
| --- | --- |
| **Value** | **Description** |
| Y | Execute commands. This is the default. |
| N | Only print commands. |

**Examples**

**A. Rebuild or reorganize all indexes with fragmentation on all user databases**

EXECUTE dbo.IndexOptimize  
@Databases = 'USER\_DATABASES',  
@FragmentationLow = NULL,  
@FragmentationMedium = 'INDEX\_REORGANIZE,INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationHigh = 'INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationLevel1 = 5,  
@FragmentationLevel2 = 30

**B. Rebuild or reorganize all indexes with fragmentation and update modified statistics on all user databases**

EXECUTE dbo.IndexOptimize  
@Databases = 'USER\_DATABASES',  
@FragmentationLow = NULL,  
@FragmentationMedium = 'INDEX\_REORGANIZE,INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationHigh = 'INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationLevel1 = 5,  
@FragmentationLevel2 = 30,  
@UpdateStatistics = 'ALL',  
@OnlyModifiedStatistics = 'Y'

**C. Update statistics on all user databases**

EXECUTE dbo.IndexOptimize  
@Databases = 'USER\_DATABASES',  
@FragmentationLow = NULL,  
@FragmentationMedium = NULL,  
@FragmentationHigh = NULL,  
@UpdateStatistics = 'ALL'

**D. Update modified statistics on all user databases**

EXECUTE dbo.IndexOptimize  
@Databases = 'USER\_DATABASES',  
@FragmentationLow = NULL,  
@FragmentationMedium = NULL,  
@FragmentationHigh = NULL,  
@UpdateStatistics = 'ALL',  
@OnlyModifiedStatistics = 'Y'

**E. Rebuild or reorganize all indexes with fragmentation on all user databases, performing sort operations in tempdb and using all available CPUs**

EXECUTE dbo.IndexOptimize @Databases = 'USER\_DATABASES',  
@FragmentationLow = NULL,  
@FragmentationMedium = 'INDEX\_REORGANIZE,INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationHigh = 'INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationLevel1 = 5,  
@FragmentationLevel2 = 30,  
@SortInTempdb = 'Y',  
@MaxDOP = 0

**F. Rebuild or reorganize all indexes with fragmentation on all user databases, using the option to maintain partitioned indexes on the partition level**

EXECUTE dbo.IndexOptimize  
@Databases = 'USER\_DATABASES',  
@FragmentationLow = NULL,  
@FragmentationMedium = 'INDEX\_REORGANIZE,INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationHigh = 'INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationLevel1 = 5,  
@FragmentationLevel2 = 30,  
@PartitionLevel = 'Y'

**G. Rebuild or reorganize all indexes with fragmentation on all user databases, with a time limit so that no commands are executed after 3600 seconds**

EXECUTE dbo.IndexOptimize  
@Databases = 'USER\_DATABASES',  
@FragmentationLow = NULL,  
@FragmentationMedium = 'INDEX\_REORGANIZE,INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationHigh = 'INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationLevel1 = 5,  
@FragmentationLevel2 = 30,  
@TimeLimit = 3600

**H. Rebuild or reorganize all indexes with fragmentation on the table Production.Product in the database AdventureWorks**

EXECUTE dbo.IndexOptimize  
@Databases = 'AdventureWorks',  
@FragmentationLow = NULL,  
@FragmentationMedium = 'INDEX\_REORGANIZE,INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationHigh = 'INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationLevel1 = 5,  
@FragmentationLevel2 = 30,  
@Indexes = 'AdventureWorks.Production.Product'

**I. Rebuild or reorganize all indexes with fragmentation except indexes on the table Production.Product in the database AdventureWorks**

EXECUTE dbo.IndexOptimize  
@Databases = 'USER\_DATABASES',  
@FragmentationLow = NULL,  
@FragmentationMedium = 'INDEX\_REORGANIZE,INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationHigh = 'INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationLevel1 = 5,  
@FragmentationLevel2 = 30,  
@Indexes = 'ALL\_INDEXES, -AdventureWorks.Production.Product'

**J. Rebuild or reorganize all indexes with fragmentation on all user databases and log the results to a table**

EXECUTE dbo.IndexOptimize  
@Databases = 'USER\_DATABASES',  
@FragmentationLow = NULL,  
@FragmentationMedium = 'INDEX\_REORGANIZE,INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationHigh = 'INDEX\_REBUILD\_ONLINE,INDEX\_REBUILD\_OFFLINE',  
@FragmentationLevel1 = 5,  
@FragmentationLevel2 = 30,  
@LogToTable = 'Y'