



Database Maintenance for Microsoft® SharePoint® Products and Technologies

Author:

Bill Baer

Date published:

February 2008

Summary:

This paper describes the recommended maintenance strategies for the databases that host content and configuration settings for SharePoint Products and Technologies.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2008 Microsoft Corporation. All rights reserved.

Microsoft, SharePoint, SQL Server, Windows, and Windows Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

.

Abstract

This white paper provides information and guidelines for maintaining the databases that host Microsoft® SharePoint® Products and Technologies data and configurations. It describes and provides examples of the database maintenance tasks that we recommend when using SharePoint Products and Technologies.

Before you implement any database maintenance tasks or modify your SharePoint Products and Technologies databases, read the following MSDN support articles:

1. [Support for changes to the databases that are used by Office server products and by Windows SharePoint Services](http://go.microsoft.com/fwlink/?LinkId=110812&clcid=0x409)
(<http://go.microsoft.com/fwlink/?LinkId=110812&clcid=0x409>)
2. [Information about the Maintenance Plan Wizard in SQL Server 2005 and about tasks that administrators can perform against SharePoint databases](http://go.microsoft.com/fwlink/?LinkId=110813&clcid=0x409)
(<http://go.microsoft.com/fwlink/?LinkId=110813&clcid=0x409>)

Table of Contents

ABSTRACT	3
INTRODUCTION	1
CHECK FOR AND REPAIR CONSISTENCY ERRORS BY USING DBCC CHECKDB	1
About DBCC CHECKDB	1
DBCC CHECKDB and performance	2
MEASURE AND REDUCE FRAGMENTATION	3
Measure fragmentation in a SQL Server 2005 database (sys.dm_db_index_physical_stats)	4
Measure fragmentation in a SQL Server 2000 database (DBCC SHOWCONTIG)	5
Reducing Fragmentation for a Database	6
Reducing fragmentation for a specific table and its indexes	9
Using ALTER INDEX	9
Fine tuning index performance by setting fill factor	10
Shrinking data files	10
Shrinking a database by using Transact-SQL commands	11
Shrinking a database by using SQL Server 2005 Management Studio	12
CREATING SQL SERVER 2005 MAINTENANCE PLANS	13
Configure a SQL Server 2005 Database Maintenance Plan	13
SUMMARY	22

Introduction

Routine database maintenance is essential for the smooth operation of Microsoft® SharePoint® Products and Technologies databases. This white paper describes the database maintenance tasks supported for SharePoint Products and Technologies databases

The recommended maintenance tasks for SharePoint Products and Technologies databases include:

- Checking database integrity.
- Defragmenting indexes by either reorganizing them or rebuilding them.
- Setting the fill factor for a server.
- Shrinking databases to recover unused disk space.

Database maintenance tasks can be performed by either executing Transact-SQL commands, or running the Database Maintenance Wizard. We will initially present the Transact-SQL commands that you can use, and then explain how to create database maintenance plans by using the Microsoft SQL Server® 2005 Database Maintenance Wizard.

Note: In this paper, we present detailed examples only for SQL Server 2005. External references for how to perform the same database maintenance tasks with SQL Server 2000 are included.

Check for and repair consistency errors by using DBCC CHECKDB

Start your routine maintenance operations with consistency checks to ensure that your data and indexes are not corrupted. You can use the DBCC (Database Console Command) CHECKDB statement to perform an internal consistency check of the data and index pages, and to repair errors.

Database consistency may be affected when: a database server is improperly shut down; a drive fails; or there are noticeable performance and availability issues. Although database consistency checks are most important after database or database server failure, a weekly database consistency check can provide important information on the health of your SharePoint Products and Technologies databases.

About DBCC CHECKDB

DBCC CHECKDB checks the logical and physical integrity of all the objects in the specified database by performing the following operations:

- Runs [DBCC CHECKALLOC](http://go.microsoft.com/fwlink/?LinkId=110815&clcid=0x409) (<http://go.microsoft.com/fwlink/?LinkId=110815&clcid=0x409>) on the database.
- Runs [DBCC CHECKTABLE](http://go.microsoft.com/fwlink/?LinkId=110833&clcid=0x409) (<http://go.microsoft.com/fwlink/?LinkId=110833&clcid=0x409>) on every table and view in the database.
- Runs [DBCC CHECKCATALOG](http://go.microsoft.com/fwlink/?LinkId=110834&clcid=0x409) (<http://go.microsoft.com/fwlink/?LinkId=110834&clcid=0x409>) on the database.
- Validates the contents of every indexed view in the database.

- Validates the Service Broker data in the database.

This means that the DBCC CHECKALLOC, DBCC CHECKTABLE, or DBCC CHECKCATALOG commands do not have to be run separately from DBCC CHECKDB. We recommend that you run DBCC CHECKDB rather than the individual operations because it identifies and repairs the widest possible errors and is generally safe to run in a production environment.

We recommend that you first run DBCC CHECKDB, and then, if it reveals errors, use DBCC CHECKDB with the REPAIR argument to repair all errors. If only one type of error is revealed, you may want to run one of the individual operations with the REPAIR argument, such as DBCC CHECKALLOC.

The following table contains sample output from DBCC CHECKDB.

```
DBCC results for 'Contoso_Content_1'.
Service Broker Msg 9675, State 1: Message Types analyzed: 14.
Service Broker Msg 9676, State 1: Service Contracts analyzed: 6.
Service Broker Msg 9667, State 1: Services analyzed: 3.
Service Broker Msg 9668, State 1: Service Queues analyzed: 3.
Service Broker Msg 9669, State 1: Conversation Endpoints analyzed: 0.
Service Broker Msg 9674, State 1: Conversation Groups analyzed: 0.
Service Broker Msg 9670, State 1: Remote Service Bindings analyzed: 0.
DBCC results for 'sys.sysrowsetcolumns'.
There are 2663 rows in 21 pages for object "sys.sysrowsetcolumns".
DBCC results for 'sys.sysrowsets'.
There are 309 rows in 4 pages for object "sys.sysrowsets".

...more

CHECKDB found 0 allocation errors and 0 consistency errors in database
'Contoso_Content_1'.
DBCC execution completed. If DBCC printed error messages, contact your
system administrator.
```

Table 1 DBCC CHECKDB Sample Output

For more information about using DBCC CHECKDB with SQL Server 2005, see [DBCC CHECKDB \(Transact-SQL\)](http://go.microsoft.com/fwlink/?LinkId=110835&clcid=0x409) (http://go.microsoft.com/fwlink/?LinkId=110835&clcid=0x409). For more information about using DBCC CHECKDB with SQL Server 2000, see [DBCC CHECKDB](http://go.microsoft.com/fwlink/?LinkId=110836&clcid=0x409) (http://go.microsoft.com/fwlink/?LinkId=110836&clcid=0x409).

DBCC CHECKDB and performance

We recommend that you run consistency checks during non-production hours, because DBCC CHECKDB acquires schema locks that prevent meta-data changes; when the TABLOCK argument is specified, DBCC CHECKDB also acquires shared table locks.

Database consistency checks on large databases can be time-consuming. If you have large databases to check, you may want to perform consistency checks on a table-by-table basis. To perform table-level consistency checks, you cannot use SQL Server 2005 or SQL Server 2000 maintenance plans, because they perform consistency checks at the database level only.

Instead, create a SQL Server Agent job that runs against the individual objects you want to check, using a command such as DBCC CHECKTABLE.

Measure and reduce fragmentation

Fragmentation occurs when the logical and physical storage allocation of a database contain many scattered areas of storage that are insufficient, not physically contiguous, or have become too fragmented to be used efficiently. Fragmentation can be the result of many inserts, updates, or deletes to a table. When a table becomes fragmented, the indexes defined on the table also become fragmented. The following figures illustrate contiguous data and fragmented data.



Figure 1 Contiguous Data (No Data Fragmentation)

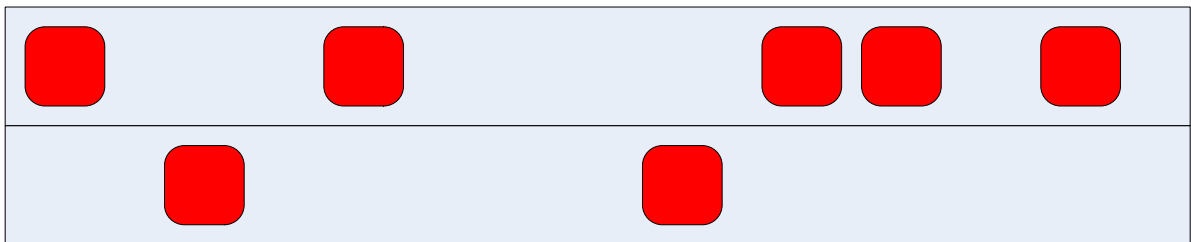


Figure 2 Fragmented Data

Because inserts, updates, and deletes are not distributed equally among the rows of the table and indexes, the fullness of each page can vary over time. Indexes fragment more rapidly than tables. For queries that scan part or all of the indexes of a table, fragmentation can cause additional page reads, which hinders parallel scanning of data and can significantly affect search performance.

Over time, database fragmentation can result in performance degradation and inefficient space utilization. To mitigate fragmentation and minimize the rate at which fragmentation occurs, manually set the size of content databases to be as large as possible given your business requirements and database architecture. For example, if you have a requirement to limit content databases to 100 gigabytes (GB), after you have created your content databases, set their size to 100 GB in SQL Server Management Studio.

Note: Although you can defragment tables, defragmenting indexes is more beneficial to database performance, and is much faster. This paper only describes how to defragment indexes. Before implementing a database fragmentation maintenance plan, you need to understand which tables and indexes are most fragmented and then create a maintenance plan to rebuild or reorganize those indexes.

In SharePoint Products and Technologies, an example of a table that often becomes fragmented is **AllDocs**, which contains document libraries, their associated documents and lists and list items, and their respective metadata.

The fragmentation level of an index is the percentage of blocks that are logically linear and physically nonlinear.

Measure fragmentation in a SQL Server 2005 database (sys.dm_db_index_physical_stats)

In SQL Server 2005, use the **sys.dm_db_index_physical_stats** dynamic management view to determine fragmentation for the indexes on specified table or view.

Notes:

- You can use the DBCC SHOWCONTIG statement with SQL Server 2005, but we do not recommend it because this feature will be removed in a future version of Microsoft SQL Server.
- If you do choose to use DBCC SHOWCONTIG, be aware that the algorithm for calculating fragmentation is more precise in sys.dm_db_index_physical_stats than in DBCC SHOWCONTIG. As a result, fragmentation values calculated by sys.dm_db_index_physical_stats appear higher. For example, in DBCC SHOWCONTIG, a table is not considered fragmented if it has page 6 and page 8 in the same extent but not page 7. However, accessing these two pages requires two physical input/output (I/O) operations, so this is counted as fragmentation sys.dm_db_index_physical_stats. For more information about using DBCC SHOWCONTIG, see [DBCC SHOWCONTIG \(Transact-SQL\)](http://go.microsoft.com/fwlink/?LinkId=110838&clcid=0x409) (<http://go.microsoft.com/fwlink/?LinkId=110838&clcid=0x409>).
- Because computers running SQL Server 2000 can only use DBCC SHOWCONTIG, if you are running both versions of SQL Server and running sys.dm_db_index_physical_stats for SQL Server 2005, you will consistently see higher fragmentation values for SQL Server 2005.

For measuring fragmentation, we recommend that you monitor the column **avg_fragmentation_in_percent**. The value for **avg_fragmentation_in_percent** should be as close to zero as possible for maximum performance. However, values from 0 percent through 10 percent may be acceptable. For more information see [sys.dm_db_index_physical_stats](http://go.microsoft.com/fwlink/?LinkId=110839&clcid=0x409) (<http://go.microsoft.com/fwlink/?LinkId=110839&clcid=0x409>)

Table 2 shows sample results from **sys.dm_db_index_physical_stats**, with a value of 9.375 for **avg_fragmentation_in_percent** in one row.

database_id	index_type_desc	alloc_unit_type_desc	avg_fragmentation_in_percent
10	CLUSTERED INDEX	IN_ROW_DATA	0
10	NONCLUSTERED INDEX	IN_ROW_DATA	0
10	NONCLUSTERED INDEX	IN_ROW_DATA	0
10	CLUSTERED INDEX	IN_ROW_DATA	0
10	NONCLUSTERED INDEX	IN_ROW_DATA	0
10	CLUSTERED INDEX	IN_ROW_DATA	0
10	NONCLUSTERED INDEX	IN_ROW_DATA	0
10	CLUSTERED INDEX	IN_ROW_DATA	0
10	NONCLUSTERED INDEX	IN_ROW_DATA	0
10	NONCLUSTERED INDEX	IN_ROW_DATA	0
10	CLUSTERED INDEX	IN_ROW_DATA	0
10	NONCLUSTERED INDEX	IN_ROW_DATA	0
10	CLUSTERED INDEX	IN_ROW_DATA	9.375

Table 2: sample results from sys.dm_db_index_physical_stats

To use the **sys.dm_db_index_physical_stats** dynamic management view

1. On the taskbar, click **Start**, point to **All Programs**, point to **Microsoft SQL Server 2005**, and then click **SQL Server Management Studio**.
To use **sys.dm_db_index_physical_stats** with a database object you must know the database ID, and object ID.
2. Select the content database in the Object Explorer, and then click **New Query**. Execute the following script.

```
SELECT DB_ID() AS [Database ID];
```

Note: When using DB_ID without specifying a database name, the compatibility level of the current database must be 90 (a SQL Server 2005 database). If you have upgraded from a previous version of SQL Server, or are using SQL Server 2000, you must specify a database name in the statement. For more information about compatibility levels, see [sp_dbcmptlevel \(Transact-SQL\)](http://go.microsoft.com/fwlink/?LinkId=110840&clcid=0x409) (<http://go.microsoft.com/fwlink/?LinkId=110840&clcid=0x409>).

3. Execute **sys.dm_db_index_physical_stats** on the database or object you have selected. You can specify not only the database, but a table or index.

Syntax:

```
sys.dm_db_index_physical_stats (
    { database_id | NULL | 0 | DEFAULT }
    , { object_id | NULL | 0 | DEFAULT }
    , { index_id | NULL | 0 | -1 | DEFAULT }
    , { partition_number | NULL | 0 | DEFAULT }
    , { mode | NULL | DEFAULT }
)
```

Measure fragmentation in a SQL Server 2000 database (DBCC SHOWCONTIG)

To check the fragmentation of database tables, an administrator can use the DBCC SHOWCONTIG function to report on logical and extent scan fragmentation. For a complete explanation of DBCC SHOWCONTIG results, see [DBCC SHOWCONTIG](http://go.microsoft.com/fwlink/?LinkId=110841&clcid=0x409) (<http://go.microsoft.com/fwlink/?LinkId=110841&clcid=0x409>).

For measuring fragmentation, we recommend you monitor the scan density value returned by DBCC SHOWCONTIG. In tables in which everything is contiguous, scan density is 100.

Table 3 shows a scan density of 20.97%, indicating that defragmentation may be beneficial.

```
DBCC SHOWCONTIG scanning 'AllDocs' table...
Table: 'AllDocs' (53575229); index ID: 1, database ID: 10
TABLE level scan performed.
- Pages Scanned.....: 104
- Extents Scanned.....: 19
- Extent Switches.....: 61
- Avg. Pages per Extent.....: 5.5
- Scan Density [Best Count:Actual Count].....: 20.97% [13:62]
- Logical Scan Fragmentation .....: 78.85%
- Extent Scan Fragmentation .....: 84.21%
```

```

- Avg. Bytes Free per Page.....: 3326.7
- Avg. Page Density (full).....: 58.90%

```

Table 32: Scan Density (Fragmented)

Table 4: shows the improved scan density after running the defragmentation script described in the following section, [Reducing Fragmentation for a Database](#).

```

DBCC SHOWCONTIG scanning 'AllDocs' table...
Table: 'AllDocs' (53575229); index ID: 1, database ID: 10
TABLE level scan performed.
- Pages Scanned.....: 64
- Extents Scanned.....: 12
- Extent Switches.....: 13
- Avg. Pages per Extent.....: 5.3
- Scan Density [Best Count:Actual Count].....: 57.14% [8:14]
- Logical Scan Fragmentation .....: 9.38%
- Extent Scan Fragmentation .....: 91.67%
- Avg. Bytes Free per Page.....: 345.9
- Avg. Page Density (full).....: 95.73%

```

Table 4: Scan Density (Defragmented)

Reducing Fragmentation for a Database

To reduce the level of index fragmentation, run the following SQL Server stored procedure. For more information about this procedure, see [How to defragment Windows SharePoint Services 3.0 databases and SharePoint Server 2007 databases](#) (<http://go.microsoft.com/fwlink/?LinkId=110843&clcid=0x409>) in the Microsoft Knowledge Base.

After determining the level of fragmentation of your databases, you can schedule the stored procedure to be run daily, weekly, or monthly depending on your needs and the overall rate of change in your environment. Generally, we recommend that you establish a weekly defragmentation schedule, at a minimum. We also recommend that you schedule defragmentation operations after running DBCC CHECKDB REPAIR operations.

```

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
IF EXISTS (SELECT * FROM sysobjects WHERE type = 'P' AND name =
'proc_DefragIndexes')
    BEGIN
        DROP Procedure dbo.proc_DefragIndexes
    END
GO

-- =====
-- This stored procedure checks all indexes in the current
-- database and performs either offline or online defragmentation
-- according to the specified thresholds.
-- The stored procedure also updates statistics for indexes in which

```

```

the last update
-- time is older than the specified threshold.
-- Parameters:
--   @onlineDefragThreshold specifies minimum percentage of
fragmentation
--   to perform online defragmentation (default 10%).
--   @offlineDefragThreshold specifies minimum percentage of
fragmentation
--   to perform offline defragmentation (default 75%).
--   @updateStatsThreshold specifies the number of days since the last
statistics update
--   which should trigger updating statistics (default 7 days).
-- =====
CREATE PROCEDURE dbo.proc_DefragIndexes
(
    @onlineDefragThreshold float = 10.0,
    @offlineDefragThreshold float = 75.0,
    @updateStatsThreshold int = 7
)
AS
BEGIN
    set nocount on
    DECLARE @objectid int
    DECLARE @indexid int
    DECLARE @frag float
    DECLARE @command varchar(8000)
    DECLARE @schemaname sysname
    DECLARE @objectname sysname
    DECLARE @indexname sysname

    declare @AllIndexes table (objectid int, indexid int,
fragmentation float)

    declare @currentDdbId int
    select @currentDdbId = DB_ID()

    insert into @AllIndexes
    SELECT
        object_id, index_id, avg_fragmentation_in_percent
    FROM sys.dm_db_index_physical_stats (@currentDdbId, NULL, NULL ,
NULL, 'LIMITED')
    WHERE index_id > 0

    DECLARE indexesToDefrag CURSOR FOR SELECT * FROM @AllIndexes

    OPEN indexesToDefrag;

    -- Loop through the partitions.
    FETCH NEXT
        FROM indexesToDefrag
        INTO @objectid, @indexid, @frag;

    WHILE @@FETCH_STATUS = 0
        BEGIN

            SELECT @schemaname = s.name

```

```

FROM sys.objects AS o
JOIN sys.schemas as s ON s.schema_id = o.schema_id
WHERE o.object_id = @objectid

SELECT @indexname = name
FROM sys.indexes
WHERE object_id = @objectid AND index_id = @indexid

IF @frag > @onlineDefragThreshold
BEGIN
    IF @frag < @offlineDefragThreshold
        BEGIN;
            SELECT @command = 'ALTER INDEX ' +
@indexname + ' ON ' + @schemaname + '.' + object_name(@objectid) + '
REORGANIZE'
                EXEC (@command)
            END;

            IF @frag >= @offlineDefragThreshold
                BEGIN;
                    SELECT @command = 'ALTER INDEX ' +
@indexname + ' ON ' + @schemaname + '.' + object_name(@objectid) + '
REBUILD'
                        EXEC (@command)
                    END;
                    PRINT 'Executed ' + @command
                END
            END

            IF STATS_DATE(@objectid, @indexid) < DATEADD(dd, -
@updateStatsThreshold, getdate())
                BEGIN
                    SELECT @command = 'UPDATE STATISTICS ' + @schemaname
+ '.' + object_name(@objectid) + ' ' + @indexname + ' WITH RESAMPLE'
                        EXEC (@command)
                    PRINT 'Executed ' + @command
                END
            END

            FETCH NEXT FROM indexesToDefrag INTO @objectid, @indexid,
@frag

        END

        CLOSE indexesToDefrag;
        DEALLOCATE indexesToDefrag;
    END
GO

```

Note: This stored procedure changes your content database indexes. Any modification to the stored procedure is not supported. For additional information on the changes that are supported for SharePoint Products and Technologies content databases see [Support for changes to the databases that are used by Office server products and by Windows SharePoint Services](http://go.microsoft.com/fwlink/?LinkId=110844&clcid=0x409) (http://go.microsoft.com/fwlink/?LinkId=110844&clcid=0x409) in the Microsoft Knowledge Base.

If the performance of a heavily fragmented database or table is not measurably improved by frequent defragmentation, you should check the database object allocation and structural integrity by using DBCC CHECKDB. For more information, see [Check for and repair consistency errors by using DBCC CHECKDB](#) earlier in this paper.

Reducing fragmentation for a specific table and its indexes

If you want to defragment the index associated with a particular table, rather than an entire database, you can either reorganize or rebuild the index.

- Reorganizing an index specifies that the index leaf level will be reorganized. Index reorganization defragments and compacts clustered and non-clustered indexes on tables and views and can significantly improve index scanning performance. Reorganization is always performed online, so that the underlying table is available to users. Reorganization is equivalent to the SQL Server 2000 DBCC INDEXDEFRAG statement.
- Rebuilding an index specifies that the index will be rebuilt using the same columns, index type, uniqueness attribute, and sort order. Rebuilding improves the performance of index scans and seeks. You can rebuild the index with a table either on or offline. Rebuilding is equivalent to the SQL Server 2000 DBCC DBREINDEX statement.

For more information, see [Clustered Index Structures](http://go.microsoft.com/fwlink/?LinkId=110847&clcid=0x409) (http://go.microsoft.com/fwlink/?LinkId=110847&clcid=0x409).

The fragmentation level of an index determines the method you should use to defragment it, and whether it can remain online, or should be taken offline.

Fragmentation level	Defragmentation method
Up to 10%	Reorganize (online)
10-75%	Rebuild (online)
75%	Rebuild (offline)

Note Using the DROP INDEX and CREATE INDEX commands is not supported on SharePoint Products and Technologies databases.

You can reorganize and rebuild indexes by using the SQL Server 2005 ALTER INDEX statement, the SQL Server 2005 Maintenance Wizard, the SQL Server 2000 DBCC INDEXDEFRAG and DBCC DBREINDEX statements, or the SQL Server 2000 Maintenance Wizard. This paper presents only the SQL Server 2005 options in detail. For more information about SQL Server 2000 options, see the following resources:

- [DBCC INDEXDEFRAG](http://go.microsoft.com/fwlink/?LinkId=111461&clcid=0x409) (http://go.microsoft.com/fwlink/?LinkId=111461&clcid=0x409)
- [DBCC DBREINDEX](http://go.microsoft.com/fwlink/?LinkId=111462&clcid=0x409) (http://go.microsoft.com/fwlink/?LinkId=111462&clcid=0x409)
- [Database Maintenance Plan Wizard](http://go.microsoft.com/fwlink/?LinkId=110849&clcid=0x409) (http://go.microsoft.com/fwlink/?LinkId=110849&clcid=0x409)

Using ALTER INDEX

ALTER INDEX permits a database administrator to perform maintenance operations against an existing table or view index. It can be used to disable, rebuild, and reorganize indexes or optionally setting options on the index. ALTER INDEX replaces the DBCC DBREINDEX and DBCC INDEXDEFRAG statements.

In most cases you can rebuild indexes while the database is online, because there are no significant gains in an offline rebuild of the indexes. However, it is important to note that when an index is being rebuilt, a shared table lock is put on the table, preventing all operations with the exception of SELECT operations from being performed. SharePoint Products and Technologies databases use clustered indexes specifically. When a clustered index is being rebuilt, an exclusive table lock is put on the table, preventing any table access by end-users.

You can customize the following sample script to rebuild all indexes on a table.

```
USE Contoso_Content_1
GO
ALTER INDEX ALL ON [database_name. [ schema_name ] . | schema_name. ]table_or_view_name
REBUILD WITH (FILLFACTOR = 70, SORT_IN_TEMPDB = ON,
STATISTICS_NORECOMPUTE = ON)
GO
```

Fine tuning index performance by setting fill factor

Fill factor can be used to further improve index data storage and performance. When indexes are created or rebuilt, the fill factor value (1-100) determines the percentage of space on each leaf level page that can be filled with data. The remaining space is reserved for future growth. For many situations the default server-wide fill factor level of 0 is optimal; however, for Microsoft Office SharePoint Server 2007, a server-wide setting of 70 is optimal to support growth and minimize fragmentation.

Although it is possible, we do not recommend that you set the fill factor for individual tables or indexes.

To view the fill factor value of one or more indexes, query the **sys.indexes** catalog view. For more information about the view, see [sys.indexes \(Transact-SQL\)](http://go.microsoft.com/fwlink/?LinkId=110850&clcid=0x409) (<http://go.microsoft.com/fwlink/?LinkId=110850&clcid=0x409>.)

To configure the server-wide fill factor value, use the **sp_configure** system stored procedure. For more information, see [spconfigure \(Transact-SQL\)](http://go.microsoft.com/fwlink/?LinkId=110851&clcid=0x409) (<http://go.microsoft.com/fwlink/?LinkId=110851&clcid=0x409>).

Shrinking data files

In SQL Server 2005 and SQL Server 2000, you can shrink each file within a database (extensions .mdf, .ldf, and .ndf) to remove unused pages and recover disk space. SharePoint Products and Technologies databases do not automatically shrink data files, although many activities create white space in the database. Activities that can create white space include running the [Stsadm mergecontentdbs operation](#), and deleting documents, document libraries, lists, list items, and sites.

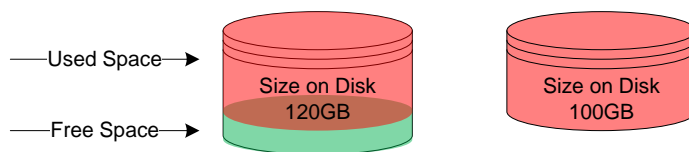


Figure 1 Database Allocation

Free space is released from the end of the file — for example, a content database file of 60 GB with a specified target size of 40 GB will free as much space as possible from the bottom 20 GB of the database file. If used pages are included in the bottom 20 GB, those pages will subsequently be relocated to the upper 40 GB of the file that is retained.

You can shrink database files individually or as a group. Shrink operations are most effective following a large file or site that has the potential to generate a large quantity of unused space. Database files can only be reduced to the point where there is no free space remaining; therefore a content database in which content is infrequently deleted may see minimal benefit from shrinking. Repeated shrinking may result in increased fragmentation because the operation does not preserve the fragmentation state of indexes. You do not need to shrink database files as

frequently as you defragment indexes. However, in environments where data is often deleted from the SharePoint Products and Technologies database, you may want to schedule database file shrinking more frequently.

Notes:

- We do not recommend that you auto-shrink databases or configure a maintenance plan that programmatically shrinks your databases.
- Shrink a database only when 50% or more of the content in it has been removed by user or administrator deletions.
- We recommend that you shrink only content databases. The configuration database, Central Administration content database, SSP databases and search databases do not usually undergo enough deletions to contain significant free space.
- Avoid the need to shrink databases by including growth allocations in your capacity planning, including an overhead allocation of 10-20%.
- Shrinking databases is a resource intensive operation. Therefore, if you must shrink a database, carefully consider when you schedule it.

Database and database files can be shrunk manually to recover space by executing the DBCC SHRINKFILE and DBCC SHRINKDATABASE statements, by using SQL Server 2005 Management Studio, or by using the SQL Server 2005 Maintenance Plan Wizard.

Shrinking a database by using Transact-SQL commands

DBCC SHRINKDATABASE shrinks the data and log files for a specific database. To shrink individual files, use DBCC SHRINKFILE.

DBCC SHRINKDATABASE

Syntax:

```
DBCC SHRINKDATABASE
( 'database_name' | database_id | 0
  [ ,target_percent ]
  [ , { NOTRUNCATE | TRUNCATEONLY } ]
)
[ WITH NO_INFOMSGS ]
```

database_name | *database_id* | **0** specifies the database name or ID. To select the current database, use 0.

target_percent is the free space in a percentage you wish to retain after the database has been shrunk.

NOTRUNCATE compacts the data in data files by moving allocated pages from the end of a file to unallocated pages in the front of the file.

TRUNCATEONLY releases all free space at the end of the file to the operating system but does not perform any page movement inside the file. Using the TRUNCATEONLY option is not recommended for SharePoint Products and Technologies content databases.

For more information, see [DBCC SHRINKDATABASE \(Transact-SQL\)](http://go.microsoft.com/fwlink/?LinkId=110852&clcid=0x409) (http://go.microsoft.com/fwlink/?LinkId=110852&clcid=0x409).

DBCC SHRINKFILE

Syntax:

```
DBCC SHRINKFILE
(
    { 'file_name' | file_id }
    { [ , EMPTYFILE ]
    | [ [ , target_size ] [ , { NOTRUNCATE | TRUNCATEONLY } ] ]
    }
)
[ WITH NO_INFOMSGS ]
```

file_name | *file_id* specifies the file name or ID.

EMPTYFILE Migrates all data from the specified file to other files in the same filegroup. **NOTE:** Using the EMPTYFILE option is not supported for SharePoint Products and Technologies database files.

target_size is the target size for the file in megabytes, expressed as an integer.

NOTRUNCATE compacts the data in data files by moving allocated pages from the end of a file to unallocated pages in the front of the file.

TRUNCATEONLY releases all free space at the end of the file to the operating system but does not perform any page movement inside the file.

Note: Using the TRUNCATEONLY option is not supported for SharePoint Products and Technologies content databases.

For more information, see [DBCC SHRINKFILE \(Transact-SQL\)](http://go.microsoft.com/fwlink/?LinkId=110853&clcid=0x409) (http://go.microsoft.com/fwlink/?LinkId=110853&clcid=0x409).

Shrinking a database by using SQL Server 2005 Management Studio

1. On the taskbar, click **Start**, point to **All Programs**, point to **Microsoft SQL Server 2005**, and then click **SQL Server Management Studio**.
2. In Object Explorer, connect to an instance of the SQL Server 2005 Database Engine and then expand that instance.
3. Expand **Databases**, right-click the database that you want to shrink, point to **Tasks**, point to **Shrink**, and then click **Files**.
4. Select the file type and file name.
5. Optionally, select **Release unused space**.

Selecting this option causes any unused space in the file to be released to the operating system and shrinks the file to the last allocated extent. This reduces the file size without moving any data.

6. Optionally, select **Reorganize files before releasing unused space**.

If you select this option, you must also set the **Shrink file to** value.

Selecting this option causes any unused space in the file to be released to the operating system and tries to relocate rows to unallocated pages.

7. Optionally, select **Empty file by migrating the data to other files in the same filegroup**.

Selecting this option moves all data from the specified file to other files in the filegroup. The empty file can then be deleted. This option is the same as executing DBCC SHRINKFILE with the EMPTYFILE option.

8. Click **OK**.

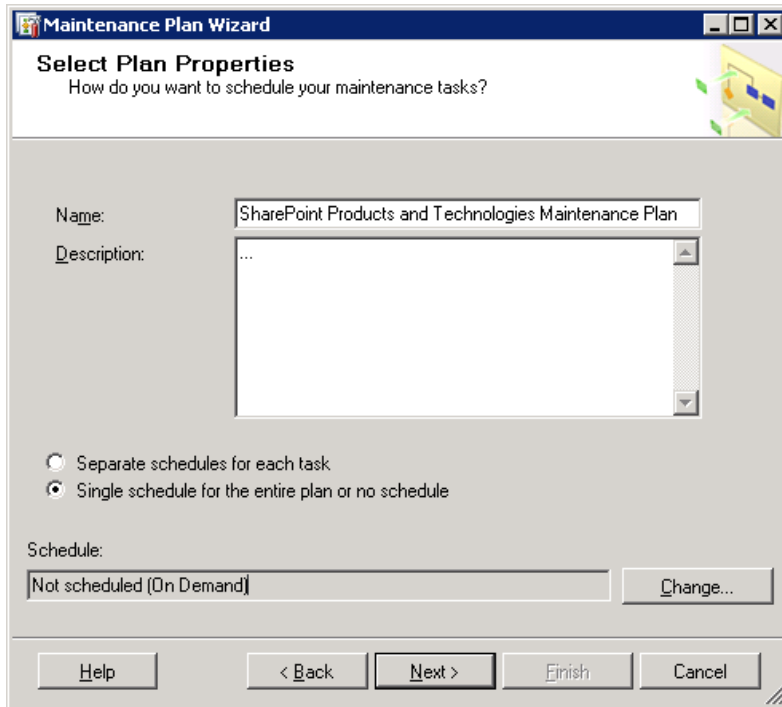
Creating SQL Server 2005 Maintenance Plans

Many of the database maintenance operations covered in this white paper can be programmatically applied through the implementation of SQL Server maintenance plans. Maintenance plans can both automate and schedule essential tasks to protect your data. By using maintenance plans in both SQL Server 2005 and SQL Server 2000, an administrator can schedule such operations as running database consistency checks, reorganizing or rebuilding indexes, and shrinking databases to reclaim unused space. For more information, see the following resources:

- [Maintenance Plan Wizard](http://go.microsoft.com/fwlink/?LinkId=110855&clcid=0x409) (http://go.microsoft.com/fwlink/?LinkId=110855&clcid=0x409) for SQL Server 2005
- [Database Maintenance Plan Wizard](http://go.microsoft.com/fwlink/?LinkId=110849&clcid=0x409) (http://go.microsoft.com/fwlink/?LinkId=110849&clcid=0x409) for SQL Server 2000

Configure a SQL Server 2005 Database Maintenance Plan

1. On the taskbar, click **Start**, point to **All Programs**, point to **Microsoft SQL Server 2005**, and then click **SQL Server Management Studio**.
2. In Object Explorer, connect to an instance of the SQL Server 2005 Database Engine and then expand that instance.
3. Click **Management**, right-click **Maintenance Plans**, and then click **Maintenance Plan Wizard**.
4. Click **Next** until you reach the Select Plan Properties page.



5. In the **Name** and **Description** fields, type a name and description.
6. Decide whether to configure one or more maintenance plans.
 - To configure a single maintenance plan, select **Single schedule for the entire plan or no schedule**.
 - To configure multiple maintenance plans with specific tasks, select **Separate schedules for each task**.

If you have an environment with 10 or more content databases or more than 250 GB of content, we recommend that you configure separate maintenance plans to provide appropriate specificity and to maximize the maintenance window.

If you configure multiple maintenance plans for a database, specify a name or description that enables you to differentiate the plans and their purposes, including their schedules.

7. Click **Change** to set a schedule for one or more plans.
The **Job Schedule Properties** dialog box appears.

Job Schedule Properties - SharePoint Products and Technologies Maintenance Plan

Name:

Schedule type: Enabled

One-time occurrence

Date: Time:

Frequency

Occurs:

Recur every: week(s) on

Monday Wednesday Friday Saturday

Tuesday Thursday Sunday

Daily frequency

Occurs once at:

Occurs every: hour(s) Starting at: Ending at:

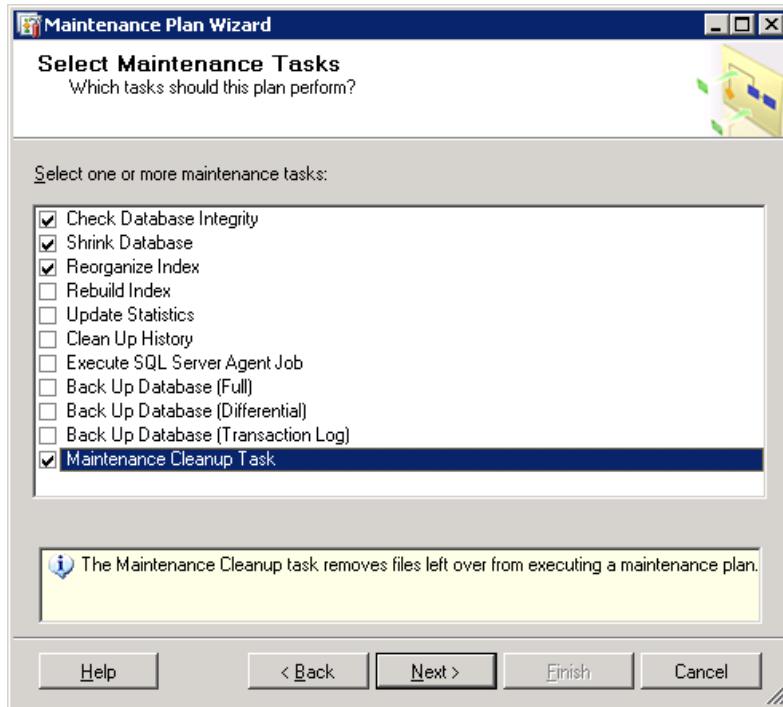
Duration

Start date: End date: No end date

Summary

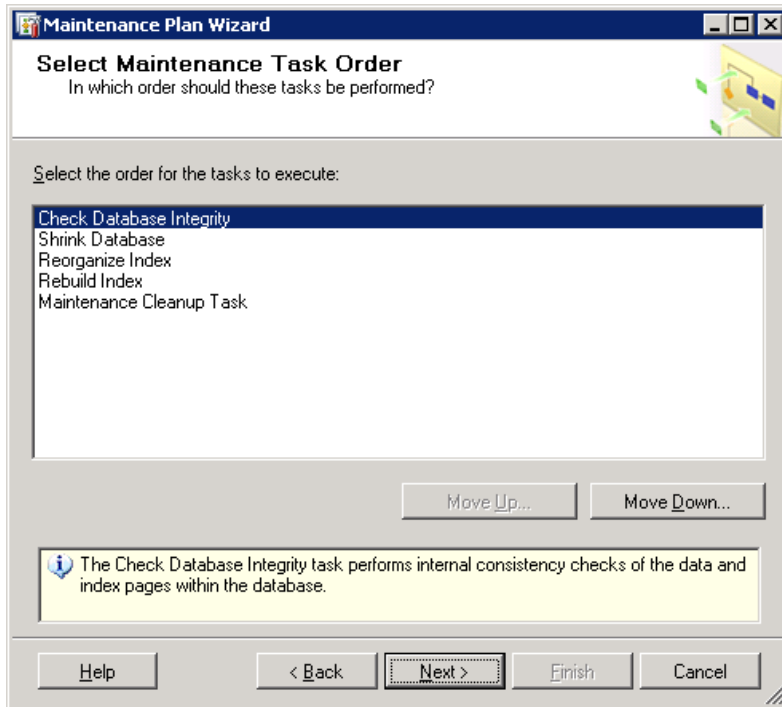
Description:

8. Complete the schedule, click **OK**, and then click **Next**.
9. On the Select Maintenance Tasks page, select the maintenance tasks to include in the plan, and then click **Next**.



Notes:

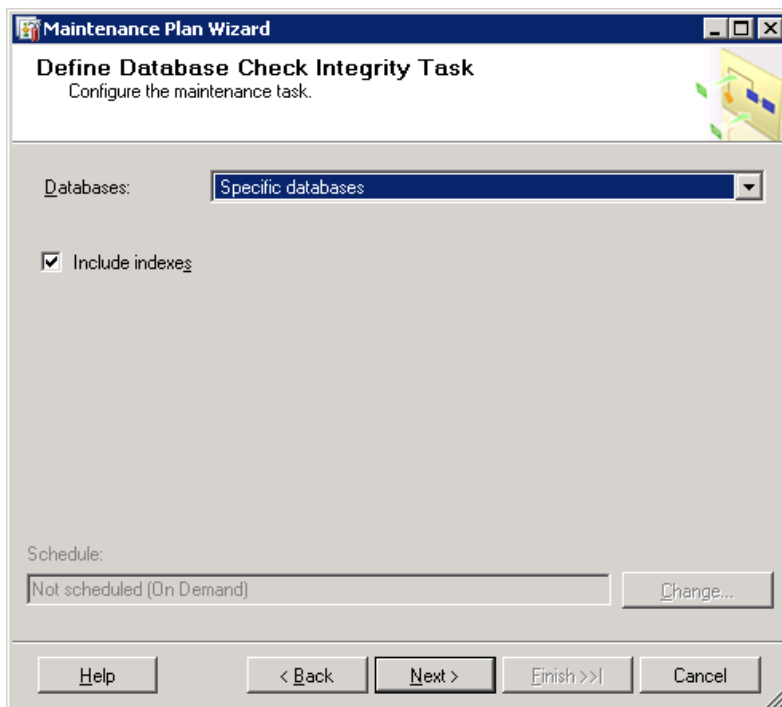
- A maintenance plan should include either index reorganization or index rebuilding; not both.
 - To determine the duration of each task, test each task individually before combining tasks into a single plan. You may need to define several maintenance plans on separate schedules to allow tasks to complete during hours when end-user operations will not be negatively affected.
 - The Maintenance Cleanup task removes files left over from executing a maintenance plan.
10. On the Select Maintenance Task Order page, change the order of the maintenance plan tasks if needed. Select a task, and then click **Move up** or **Move down**. When tasks are correctly ordered, click **Next**.
- Note:** Always begin with the database integrity check. If the integrity check fails, do not perform the remaining tasks. Instead, repair the suspect database.



Next, the wizard guides you through setting the details for each task.

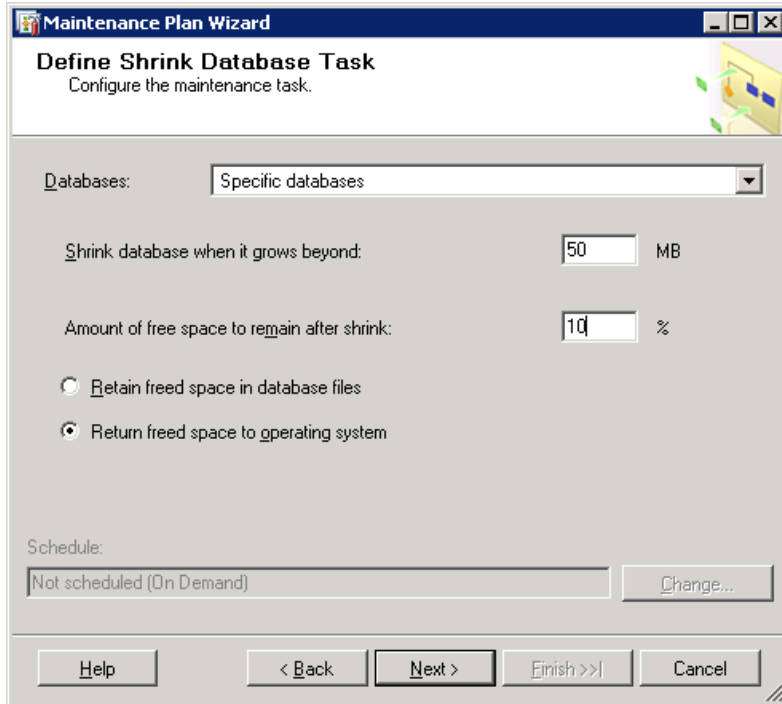
11. On the Define Database Check Integrity Task page, select the databases to check for integrity, and then click **Next**.

Note: You can safely check all SharePoint Products and Technologies databases for integrity.

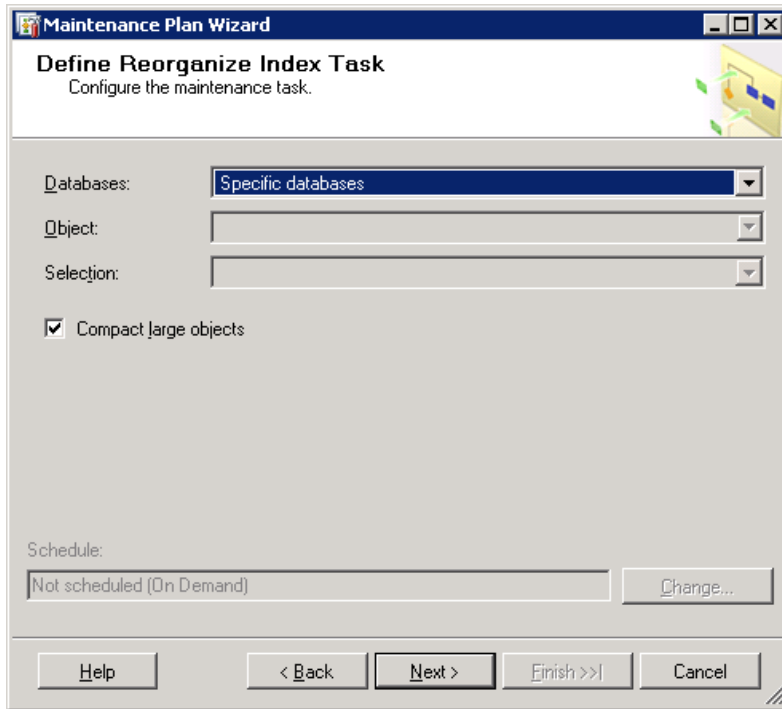


12. On the Define Shrink Database Task page, in the **Databases** list, select databases to shrink.

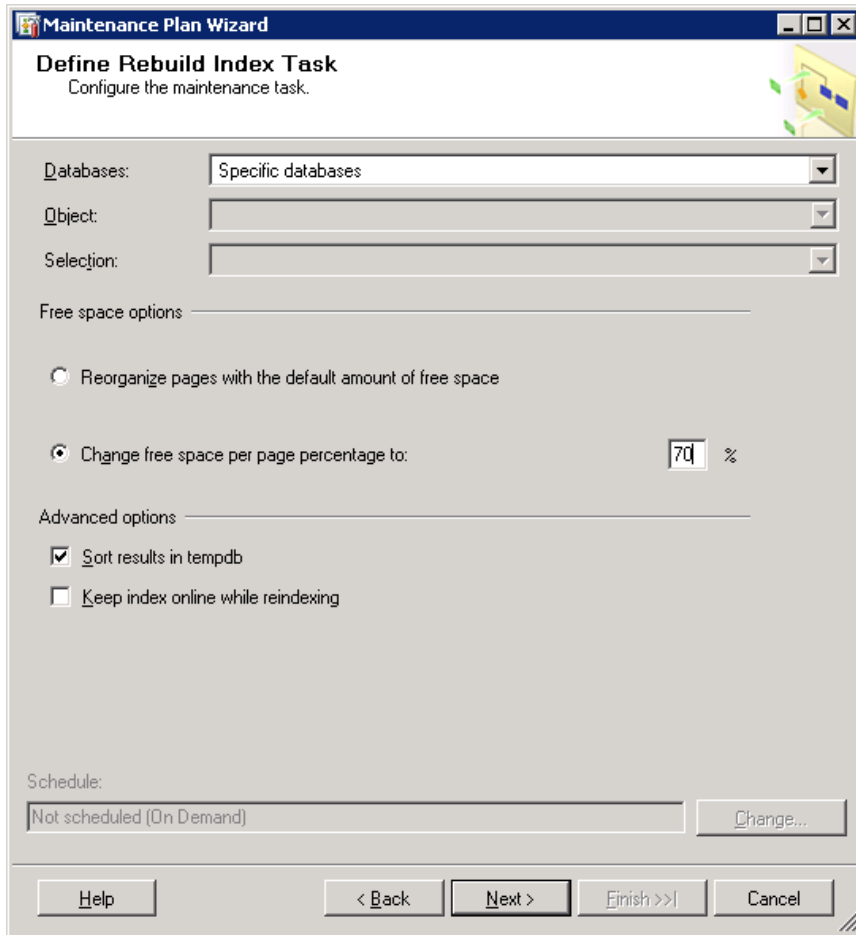
Note: To maintain a sustainable and stable SharePoint Products and Technologies deployment, you should only shrink content databases because deletion are most common in these databases, Shrinking the configuration database, Central Administration content database, SSP databases, and Search databases is unnecessary and can lead to fragmentation.



13. In the **Shrink database when it grows beyond** field, type the size in MB at which a database should be a candidate for shrinking.
Set this value to 20% more than the maximum size you want your content databases to grow. For example, if you have established a database architecture that allows up to 100 GB for each content database, set this value to 120 GB.
14. In the **Amount of free space to remain after shrink** box, type the percentage of free space to retain in each database after the shrink operation.
We recommend that you specify 10% for this value. Setting this value can help reduce fragmentation when you have scheduled frequent shrinking.
15. Select either **Retain freed space in database files** or **Return freed space to operating system**, and then click **Next**.
We recommend that you return the freed space to the operating system to reduce instances of failures in database expansion, creation of new databases, and search and indexing operations.
16. On the Define Reorganize Index Task page, in the **Databases** list, specify the databases to reorganize the indexes for, select the **Compact large objects** check box, and then click **Next**.



17. On the Define Rebuild Index Task page, in the **Databases** list, specify the databases to reorganize the indexes for.
18. Select **Change free space per page percentage**, type 70, and then click **Next**.
Change free space per percentage sets the fill factor for the database.



19. On the Define Maintenance Cleanup Task page, set the values that meet your needs, and then click **Next**.

We recommend that you delete Maintenance Plan text reports.

Maintenance Plan Wizard

Define Maintenance Cleanup Task

Configure the maintenance task.

Delete files of the following type:

- Backup files
- Maintenance Plan text reports

File location:

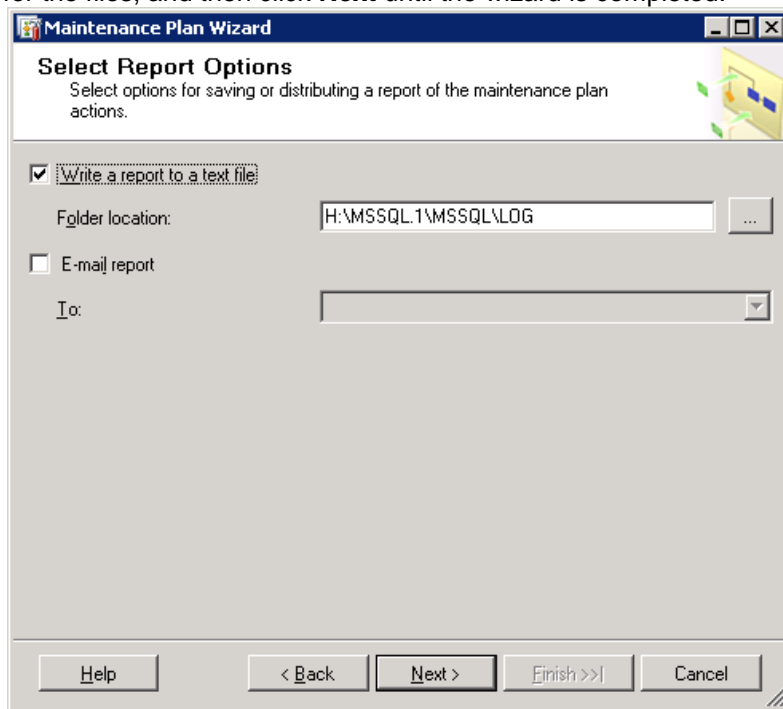
- Delete specific file
- File name: - Search folder and delete files based on an extension
- Folder: - File extension:
- Include first-level subfolders

File age:

- Delete files based on the age of the file at task run time
- Delete files older than the following:

Schedule:

20. On the Select Report Options page, select **Write a report to a text file**, select a location for the files, and then click **Next** until the wizard is completed.



Summary

Whichever method you choose to use, consistently maintaining the databases that host SharePoint Products and Technologies can significantly improve the health and performance of your system.

Ensure that you have reliable backups for all databases before you implement maintenance operations and maintenance plans.

Before you implement consistently running maintenance operations or a maintenance plan, test the impact of the operations on your system, and the time required to run them.

As much as possible, set any maintenance operations or maintenance plans to run during off hours to minimize the performance impact to users.