

Implementing Virtual Provisioning on EMC Symmetrix with Microsoft SQL Server

Applied Technology

Abstract

This white paper provides a detailed review of the technical aspects and benefits of deploying Microsoft SQL Server databases on EMC[®] Symmetrix[®] arrays using Virtual Provisioning[™].

December 2009

Copyright © 2008, 2009 EMC Corporation. All rights reserved.

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED “AS IS.” EMC CORPORATION MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

For the most up-to-date listing of EMC product names, see EMC Corporation Trademarks on EMC.com

All other trademarks used herein are the property of their respective owners.

Part Number h4287.2

Table of Contents

Executive summary	4
Introduction	4
Audience	4
Terminology	4
Symmetrix and thin devices	6
Requirements	7
Windows NTFS considerations	7
Windows partition alignment.....	8
Windows NTFS volume format.....	8
Windows Server 2003 volume format	9
Windows Server 2008 volume format	10
Deploying Microsoft SQL Server files on thin devices.....	10
Transaction log files	10
SQL Server Instant File Initialization.....	11
SQL Server TEMPDB database	13
Using Microsoft SQL Server Auto-Grow and Virtual Provisioning	14
Space reclamation	14
Space reclamation considerations for SQL Server.....	15
Transaction log files	15
Database files.....	15
Local and remote replication between thin devices	17
Local replication between thin and regular devices	17
Performance considerations	17
Automated pool rebalancing	17
Thin pool management.....	24
Thin pool monitoring	24
Exhaustion of oversubscribed pools	25
Conclusion	27
References	27
Appendix: Virtual Provisioning examples	29
Create a thin pool.....	29
Add data devices to a thin pool.....	29
Bind thin devices to a pool	30

Executive summary

EMC® Symmetrix® DMX-3 and DMX-4 and Symmetrix V-Max™ storage arrays continue to provide increased storage utilization and optimization, enhanced capabilities, and greater interoperability and security, as well as multiple ease-of-use improvements. One such feature available with Engenuity™ release 5773 and later that provides increased storage utilization and optimization is Symmetrix Virtual Provisioning™.

Virtual Provisioning, generally known in the industry as “thin provisioning,” enables organizations to improve ease of use, enhance performance, and increase capacity utilization for certain applications and workloads. The implementation of Virtual Provisioning for Symmetrix DMX-3 and DMX-4 and Symmetrix V-Max storage arrays directly addresses improvements in storage infrastructure utilization, as well as associated operational requirements and efficiencies.

This white paper addresses the considerations for deploying a Microsoft SQL Server 2005 or 2008 database on thinly provisioned devices. An understanding of the principles that are exposed here will allow the reader to deploy Microsoft SQL Server databases with Virtual Provisioning in the most effective manner.

Introduction

One of the biggest challenges facing storage administrators is storage provisioning for new applications. Administrators typically allocate space based on anticipated future growth of applications. This is done to mitigate recurring operational functions, such as incrementally increasing storage allocations or adding discrete blocks of storage as existing space is consumed. Using this approach results in more physical storage being allocated to the application than is needed for a significant amount of time and at a higher initial cost than is necessary. This overprovisioning of physical storage also leads to increased power, cooling, and floor space requirements. Even with the most careful planning, it may be necessary to provision additional storage in the future, which could potentially require an application outage.

A second layer of storage overprovisioning happens when a database administrator overallocates storage for a file or filegroup to ensure sufficient free space within the database. The operating system sees the space as completely allocated but internally only a fraction of the allocated space might be used.

EMC Virtual Provisioning can address both of these issues. Virtual Provisioning allows more storage to be presented to a host operating system than is physically available. More importantly, Virtual Provisioning will allocate physical storage only when the storage is actually written to. This allows more flexibility in predicting future growth and reduces the initial costs of provisioning storage to an application, and it can obviate the inherent waste in overallocation of space and administrative management of subsequent storage allocations.

Audience

This white paper is intended for storage architects and administrators, and database administrators responsible for deploying Microsoft SQL Server 2005 or 2008 on Symmetrix DMX-3 and DMX-4 and Symmetrix V-Max with Virtual Provisioning.

Terminology

Table 1. Basic Symmetrix array terms

Term	Description
Device	A logical unit of storage defined within a Symmetrix storage unit.
Device Capacity	The actual storage capacity of a device.

Device Extent	The size of the smallest contiguous region of a device for which an extent mapping can occur.
Host Accessible Device	A device that is exported for host use.
Internal Device	A device used for an internal function of the storage unit.
Metadevice	An aggregation of host accessible devices seen from the host as a single device.
Pool	A collection of internal devices for some specific purpose.

Table 2. Virtual Provisioning terms

Term	Description
Thin Device	A host accessible device that has no storage directly associated with it.
Data Device	An internal device that provides storage capacity to be used by thin devices.
Thin Device Extent	The minimum quantum of storage that must be mapped at a time to a thin device.
Data Device Extent	The minimum quantum of storage that is allocated at a time when dedicating storage from a thin pool for use with a specific thin device.
Extent Mapping	Specifies the relationship between the thin device and data device extents.
Thin Pool	A collection of data devices that provide storage capacity for thin devices.
Thin Pool Capacity	The sum of the capacities of the member data devices.
Bind	The process by which one or more thin devices are associated to a thin pool.
Unbind	The process by which a thin device is diassociated from a given thin pool. When unbound, all previous extent allocations from the data devices are erased and returned for reuse.
Enabled Data Device	A data device belonging to a thin pool on which extents can be allocated for thin devices bound to that thin pool.
Disabled Data Device	A data device belonging to a thin pool from which capacity cannot be allocated for thin devices. This state is under user control.
Thin Pool Enabled Capacity	The sum of the capacities of enabled data devices belonging to a thin pool.
Thin Pool Allocated Capacity	A subset of thin pool enabled capacity that has been allocated for the exclusive use of all thin devices bound to that thin pool.
Thin Device Written Capacity	The capacity on a thin device that was written to by a host. In most implementations this is a subset of the thin device allocated capacity.
Oversubscribed Thin Pool	A thin pool whose thin pool enabled capacity is less than the sum of the reported sizes of the thin devices using the pool.
Thin Device Subscription Ratio	The ratio between the thin device subscribed capacity and the thin pool enabled capacity. This value is expressed as a percentage.
Thin Device Utilization Ratio	The ratio between the thin device written capacity and thin device allocated capacity. This value is expressed as a percentage.
Thin Pool Subscription Ratio	The ratio between the sum of the thin device subscribed capacity of all its bound thin devices and the associated thin pool enabled capacity. This value is expressed as a percentage.
Thin Pool Allocation Ratio	The ratio between the thin device allocated capacity and thin pool enabled capacity. This value is expressed as a percentage.
Pre-Provisioning	Allocating a range of extents from the thin pool to the thin device at the time the thin device is bound. Sometimes used to reduce the operational impact of allocating extents or to eliminate the potential for a thin device to run out of available extents

Symmetrix and thin devices

Symmetrix thin devices are logical devices that can be used in many of the same ways that Symmetrix devices have traditionally been used. Unlike traditional Symmetrix devices, thin devices do not need to have physical storage completely allocated at the time the device is created and presented to a host. The physical storage that is used to supply disk space to thin devices comes from a shared storage pool called a thin pool that the thin device has been previously bound to. Multiple thin devices can be bound to any given thin pool. The thin pool itself is comprised of devices called data devices that provide the actual physical storage to support the thin device allocations.

When a write is performed to a part of any bound thin device for which physical storage has not yet been allocated, the Symmetrix allocates physical storage from the thin pool for that portion of the thin device only. The Symmetrix operating environment, Enginuity, satisfies the requirement by providing a block of storage from the thin pool called a thin device extent. This approach reduces the amount of storage that is actually consumed.

The minimum amount of physical storage that can be reserved at a time for the dedicated use of a thin device is referred to as the data device extent. An entire data device extent is physically allocated to the thin device at the time the thin storage allocation is made as a result of a host write operation. The data device extent is allocated from any one of the data devices in the associated thin pool. Allocations across the data devices are balanced to ensure that an even distribution of allocations occurs from all available data devices. This results in distributing I/O workload across the data devices.

The minimum amount of physical storage that must be mapped to a thin device at a time is called a thin device extent. For Symmetrix, the thin device extent size is the same as the data device extent size.

When a read is performed on a thin device, the data being read is retrieved from the appropriate data device in the thin pool that contains the extent allocation. If for some reason a read is performed against an unallocated portion of the thin device, zeroes are returned to the reading process.

When more physical data storage is required to service existing or future thin devices, for example, when a thin pool is approaching maximum allocated capacity, data devices can be added to existing thin pools dynamically without needing a system outage. New thin devices can also be created and bound with existing thin pools. A thin device can only be bound to a single thin pool at any given time.

When data devices are added to a thin pool they can be in an enabled or disabled state. In order for the data device to be used for thin extent allocation it needs to be in the enabled state. For it to be removed from the thin pool, it needs to be in a disabled state. Depending on the Enginuity level, disabling data devices with allocated extents will cause the data devices to drain and copy any allocated extents to the remaining, enabled devices in the pool. Therefore, data devices can be removed from a pool and repurposed without having to unbind thin devices that may have allocated extents associated with the data devices. The following figure depicts the relationships between thin devices and their associated thin pools. There are nine devices associated with thin Pool A and three thin devices associated with thin pool B.

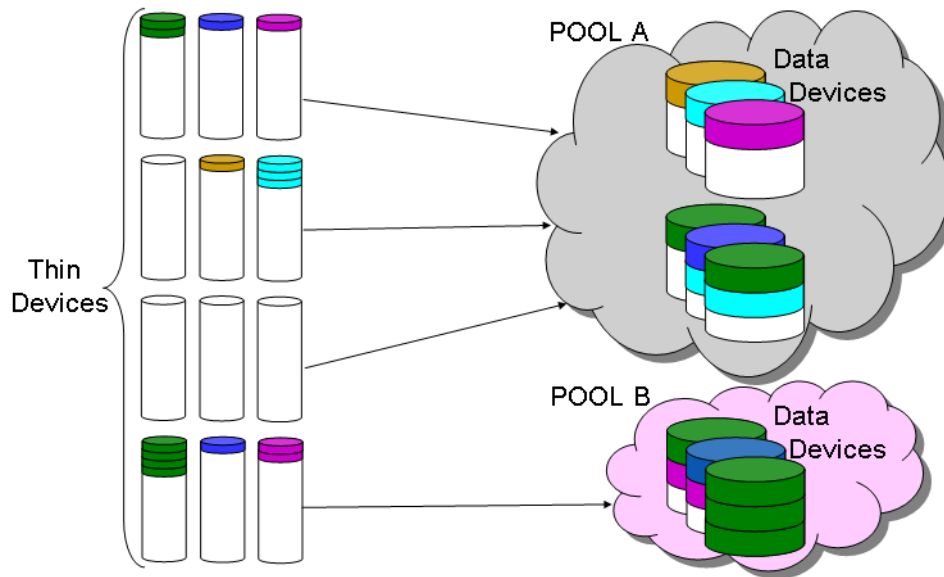


Figure 1. Thin devices and thin pools containing data devices

The way data extents are allocated across the data devices results in a form of striping in the thin pool. The more data devices that are in the thin pool, then the wider the striping is and the greater the number of devices that can participate in application I/O. The data extent size is 768 KB. The maximum size of a thin device for Enginuity 5773 is 60 GB. The maximum size of a thin device for Enginuity 5874 is 240 GB. If a larger size is needed, then a metavolume comprised of thin devices can be created. It is recommended that the metavolume be concatenated rather than striped since the thin pool is already striped using thin extents. Concatenated metavolumes also support fast expansion capabilities, as new metavolume members can easily be appended to the existing concatenated metavolume. This functionality may be applicable when the provisioned thin device has become fully allocated at the host level, and it is required to further increase the thin device to gain additional space.

Striped metavolumes are supported with Virtual Provisioning and there may be workloads that will benefit from multiple levels of striping.

Requirements

Virtual Provisioning requires an Enginuity code level of 5773 or later. In order to create thin pools and thin devices and manage them, Solutions Enabler 6.5.0 or later is required. If the Symmetrix Management Console (SMC) is being used to manage Virtual Provisioning components, version 6.1 or later is needed. Thin pools can only be created by the customer and cannot be created during the bin file (Symmetrix configuration) creation process.

Depending on the version of Enginuity there may be restrictions on RAID protections, replication options, and feature functionality available when using Virtual Provisioning. For more detailed information on Virtual Provisioning, including available functionality dependent on Enginuity, please see the technical note *Best Practices for Fast, Simple Capacity Allocation with EMC Symmetrix Virtual Provisioning* available on Powerlink®.

Windows NTFS considerations

Microsoft SQL Server 2005 or 2008 data files and transaction logs typically exist on Windows Server NTFS volumes. Therefore there is a requirement to understand the recommendations and behavior of Windows Server NTFS volume creation and administration for LUNs comprised of EMC Symmetrix Virtual Provisioning devices.

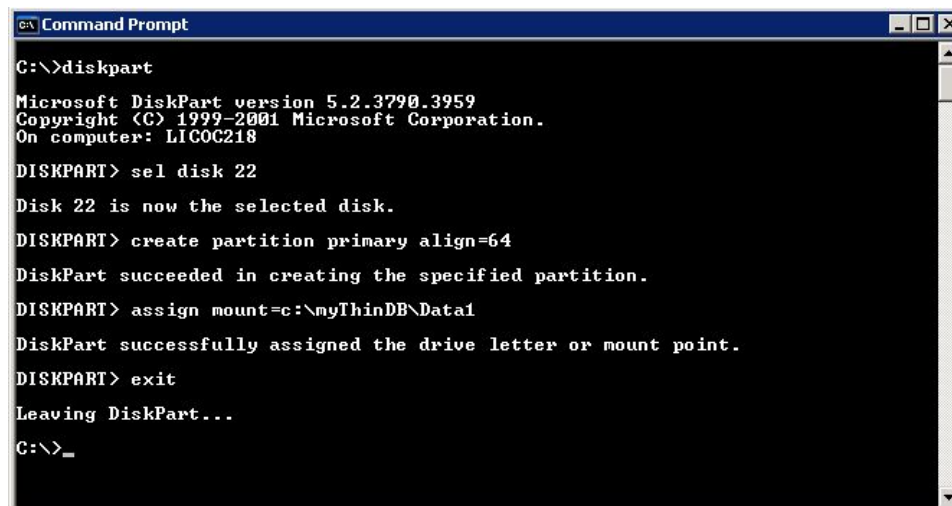
Windows Server 2003 and later versions include NTFS implementations that behave in a “thin friendly” manner. Creation of the NTFS volume and subsequent format operations will only allocate a small number of the total extents represented by the thin devices.

Previous recommendations for creation, partition alignment, and NTFS allocation unit size select for volumes that will contain Microsoft SQL Server 2005 or 2008 data and transaction log files are applicable.

Windows partition alignment

Disk devices created from Symmetrix thin devices need to be managed in the same manner as traditional devices. This includes all recommendations, including correctly aligning disk partitions. All LUNs used to host Microsoft SQL Server database data and transaction log files should be correctly aligned to provide optimal performance.

Figure 2 demonstrates the use of the DISKPART utility under Windows Server 2003 SP2. The execution of the “create partition primary align=64” statement will cause the creation of an aligned volume that will represent the entire basic disk.



```
C:\>diskpart
Microsoft DiskPart version 5.2.3790.3959
Copyright (C) 1999-2001 Microsoft Corporation.
On computer: LIC0C218

DISKPART> sel disk 22
Disk 22 is now the selected disk.

DISKPART> create partition primary align=64
DiskPart succeeded in creating the specified partition.

DISKPART> assign mount=c:\myThinDB\Data1
DiskPart successfully assigned the drive letter or mount point.

DISKPART> exit
Leaving DiskPart...
C:\>_
```

Figure 2. Windows partition alignment operation

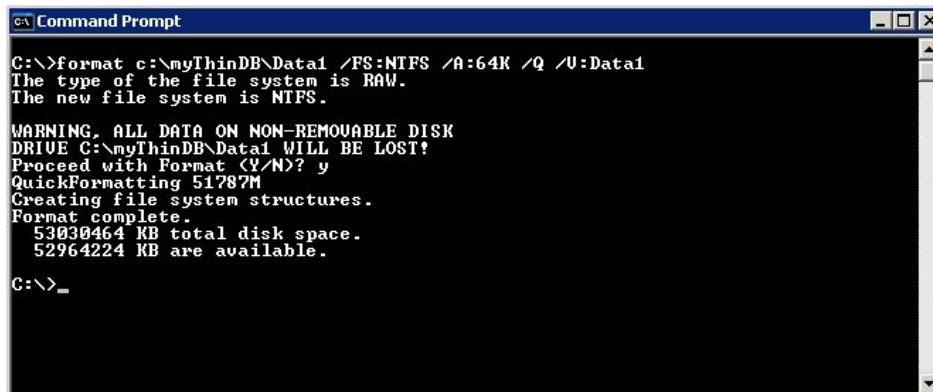
GUID Partition Table (GPT) disk devices can be created in a similar manner. Further details on the disk partition alignment operations can be found on EMC Powerlink in the white paper *Using diskpar and diskpart to Align Partitions on Windows Basic and Dynamic Disks*.

Windows Server 2008 automatically creates correctly aligned partitions, and no user override is required.

The final statements in the DISKPART session mount the volume at the C:\myThinDB\Data1 location. This represents a Windows junction point, also known as a mount point. Mount points are typically used to mount volumes to avoid the limitations imposed by the limited number of drive letters. Customers may choose the appropriate mechanism to mount the volumes as there is no performance difference between the use of drive letters or mount points for Microsoft SQL Server database file or transaction log locations.

Windows NTFS volume format

Once volumes have been created on the thin devices presented to a Windows server as LUNs, it is necessary to format the volume as Windows NTFS. The example shown in Figure 3 details a command line execution of the FORMAT . EXE command. It is both a Microsoft SQL Server and EMC recommendation to use an NTFS Allocation Unit (AU) size of 64 KB. In the example, this is represented by the /A:64K option. The volume used for the Data1 mount point can be seen to have 52 GB of usable storage allocation.



```
c:\>format c:\myThinDB\Data1 /FS:NTFS /A:64K /Q /U>Data1
The type of the file system is RAW.
The new file system is NTFS.

WARNING. ALL DATA ON NON-REMOVABLE DISK
DRIVE C:\myThinDB\Data1 WILL BE LOST!
Proceed with Format (Y/N)? y
QuickFormatting 51787M
Creating file system structures.
Format complete.
 53030464 KB total disk space.
 52964224 KB are available.

C:\>_
```

Figure 3. Windows NTFS volume format

Windows Server 2003 volume format

It is also possible to use the quick format option /Q to reduce the actual time and allocations created by the Windows NTFS format. For Windows 2003, however, the use of this command line option is not mandatory. Even when a format command is executed without the use of the quick format option, the actual allocations do not differ. The “non quick” format operation simply adds an additional scan of the disk device in search of media errors. This read activity will not generate any additional thin pool allocations to occur. For example, Figure 4 shows the thin pool allocations after the three thin devices were formatted. In the display, the thin devices, which represent the LUNs accessed by the Windows server, are listed under the Thin Devices section. These three Symmetrix devices (0432, 0435, and 0438) represent the volumes mounted at C:\myThinDB\Data1, C:\myThinDB\Data2, and C:\myThinDB\Log, respectively.

During the format phase the NTFS quick format option was used for the Data1 (0432) and Log (0438) volumes. The volume representing Data2 (0435) did not use the quick option. Data1 and Data2 are thin devices of the same size and the Log device is physically smaller. When comparing the allocations of Data1 and Data2 it is clear that both thin devices have the same number of both Pool Allocated Tracks and Pool Written Tracks. Thus not specifying the quick option will not result in a fully allocated thin device.

With Windows Server 2008 the behavior of a “non-quick” NTFS volume format changes significantly. A non-quick format will result in a fully allocated thin device. Please see the “Windows Server 2008 volume format” section for more details

```

c:\>symcfg -sid 536 show -pool SQL_DB1 -thin -detail
Symmetrix ID: 000190300536
Symmetrix ID      : 000190300536
Pool Name         : SQL_DB1
Pool Type         : Thin
Dev Emulation     : FBA
Dev Configuration : 2-Way Mir
Pool State        : Enabled
# of Devices in Pool : 5
# of Enabled Devices in Pool : 5
Max. Subscription Percent : None

Enabled Devices(5):
<
-----
Sym Dev      Total Tracks  Alloc Tracks  Free Tracks  Full (%)  Device State
-----
04CE        276204      660      275544      0      Enabled
04CF        276204      684      275520      0      Enabled
04D0        276204      672      275532      0      Enabled
04D1        276204      684      275520      0      Enabled
04D2        276204      660      275544      0      Enabled
Tracks      1381020      3360      1377660      0
>

Thin Devices(3):
<
-----
Sym Dev      Total Tracks  Pool Subs (%)  Pool Allocated Tracks  Pool Written Tracks  Pool Written (%)  Status
-----
0432        828630      60      1128      0      1043      0      Bound
0435        828630      60      1128      0      1043      0      Bound
0438        276210      20      1104      0      1040      0      Bound
Tracks      1933470      140      3360      0      3126      0
>

```

Figure 4. Thin pool display after NTFS format operations

It should also be noted in Figure 4 that a somewhat even distribution of thin extent allocations is occurring from the data devices allocated to the pool. This is shown in the Alloc Tracks column in the Enabled Devices section. Each of the five data devices has an allocation of tracks that is somewhat even, based on the actual write operations occurring against the thin devices.

Windows Server 2008 volume format

Windows Server 2008 introduces a significant change in the behavior of a NTFS volume format where the quick option is not used. When a non-quick format is selected, every block on the file system will be overwritten with zeroes. From a Virtual Provisioning perspective this will cause a thin device to become fully allocated within its respective thin pool. With this behavior in mind, it is important to select the quick format option /Q when formatting any thin device on Windows Server 2008. A quick format will perform similarly to Windows 2003 where only a small number of tracks will become allocated within the thin pool.

Deploying Microsoft SQL Server files on thin devices

There are certain considerations that a database administrator must be aware of when deploying a Microsoft SQL Server 2005 or 2008 database using Virtual Provisioning. How the various data and transaction log files as well as some other components of the SQL Server environment work in a Virtual Provisioning environment depends on the component itself and its usage patterns. This section describes how the various Microsoft SQL Server components interact with Virtual Provisioning devices.

Transaction log files

Active log files are formatted when a Microsoft SQL Server 2005 or 2008 database is created. Every single page of the log files is written to at this time so that the log files become fully provisioned when they are initialized and will not cause any thin extent allocations after this.

It is important to understand that due to the format operation, the active log files become striped across all the devices in the thin pool, using the stripe extent size as detailed in the “Symmetrix and thin devices” section. Therefore, no single physical disk will incur the overhead of all the writes generated to the transaction log files. If the requirements for a particular implementation require that the transaction logs need to be separated from the data files, this can be achieved by creating a second thin pool that is dedicated for the active log files.

SQL Server Instant File Initialization

Microsoft SQL Server 2005 introduced new functionality that altered the behavior during a database creation phase, and this functionality is also available in SQL Server 2008. Prior to SQL Server 2005, earlier versions of SQL Server would fully initialize all data file and transaction log file components, writing to every page in all database files and transaction log files. As of SQL Server 2005, the database creation phase was no longer required to initialize all data files if Instant File Initialization could be utilized by the SQL Server environment.

EMC recommends deployment of Microsoft SQL Server 2005 (and later) with Instant File Initialization functionality with EMC Symmetrix Virtual Provisioning. Earlier versions of SQL Server that do not include Instant File Initialization will result in full extent allocations for the physical files located on thin devices. This will result from SQL Server writing to every data page. Details of Microsoft SQL Server Instant File Initialization can be found in the *Microsoft SQL Server 2005 Books Online* documentation set that is provided on the installation media.

Transaction log files will be fully allocated even when using Instant File Initialization.

Figure 5 shows the database create operation using mount point locations that are located on thin devices. The T-SQL sample CREATE DATABASE statement for the myThinDB database does not require any special options when using thin devices.

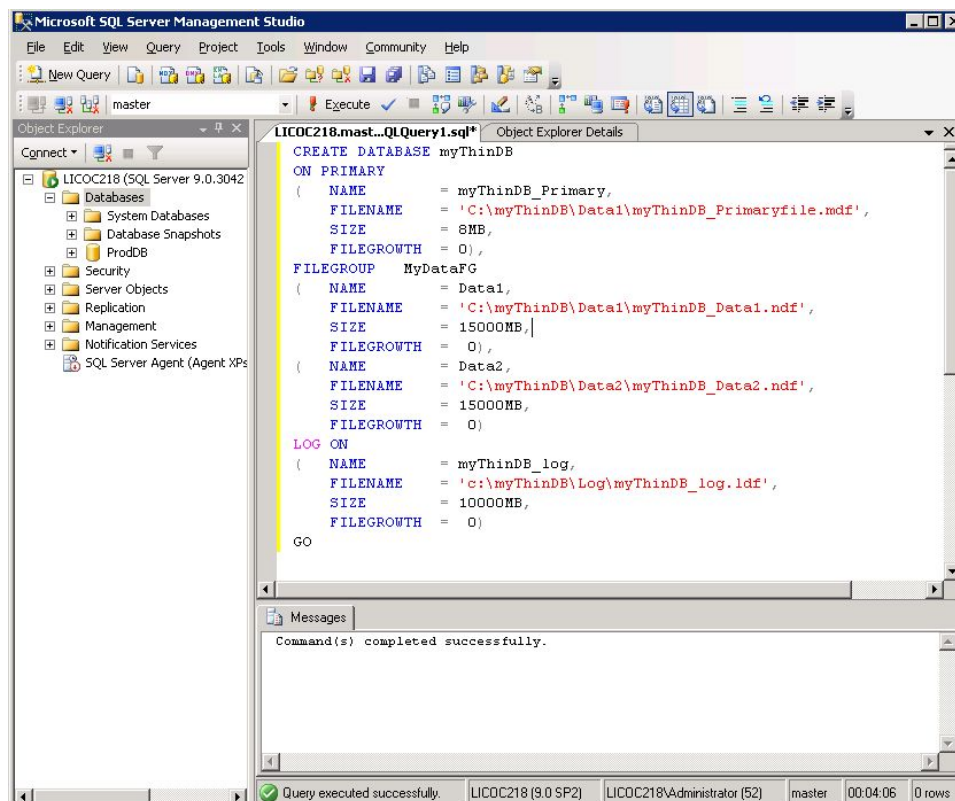


Figure 5. Creation of a SQL Server database

It should be noted that the T-SQL statement will result in the creation of four physical files. The first is the primary file group, which in this instance is configured with only 8 MB of space. Two data files, each of 15 GB, are also defined to be created. The data files myThinDB_Data1.ndf and myThinDB_Data2.ndf are located on the two mounts point previously defined. Finally, the transaction log is defined to be 10 GB in size, and is located on the mount point prepared for the transaction log.

After the execution of the CREATE DATABASE statement, a display of the thin pool devices and associated allocations is shown in Figure 6. Again, Symmetrix devices 0432 and 0435 shown in the Thin Devices section represent the data file locations for myThinDB_Data1.ndf and myThinDB_Data2.ndf. Symmetrix device 0438 is the location for the transaction log file myThinDB_Log.ldf.

```

C:\>symcfg -sid 536 show -pool SQL_DBI -thin -detail
Symmetrix ID: 000190300536

Symmetrix ID      : 000190300536
Pool Name         : SQL_DBI
Pool Type        : Thin
Dev Emulation     : FBA
Dev Configuration : 2-Way Mir
Pool State        : Enabled
# of Devices in Pool : 5
# of Enabled Devices in Pool : 5
Max. Subscription Percent : None

Enabled Devices(5):
<
-----
Sym      Total      Alloc      Free Full  Device
Dev      Tracks     Tracks     Tracks <%>  State
-----
04CE     276204     33792     242412   12   Enabled
04CF     276204     33720     242484   12   Enabled
04D0     276204     33936     242268   12   Enabled
04D1     276204     33960     242244   12   Enabled
04D2     276204     33744     242460   12   Enabled
-----
Tracks   1381020     169152     1211868   12
>

Thin Devices(3):
<
-----
Sym      Total      Pool      Pool      Pool
Dev      Tracks     Subs     Allocated  Written
          Tracks     (%)      Tracks     (%)      Tracks     (%)  Status
-----
0432     828630      60       4044      0       1304      0   Bound
0435     828630      60       4020      0       1284      0   Bound
0438     276210      20      161088     12     161040     12   Bound
-----
Tracks   1933470     140      169152     12     163628     12
>
C:\>

```

Figure 6. Thin pool display after database creation

It is noticeable that the Pool Allocated Tracks and Pool Written Tracks for the transaction log device 0438 counters are substantially higher than for the data file locations. This is a result of Microsoft SQL Server writing every page of the defined transaction log, which is 10 GB in size. It is clear, therefore, that the data files, which are defined to be 15 GB each, are not fully allocated and that SQL Server 2005 Instant File Initialization is recommended for Symmetrix Virtual Provisioning. This functionality allows customers to fully leverage the benefits offered by the technology.

When using Microsoft SQL Server 2005 Management Studio to view the resulting database, as shown in Figure 7, it should be noted that SQL Server is unaffected by the use of thin devices, and shows full allocation of devices as defined in the CREATE DATABASE statement.

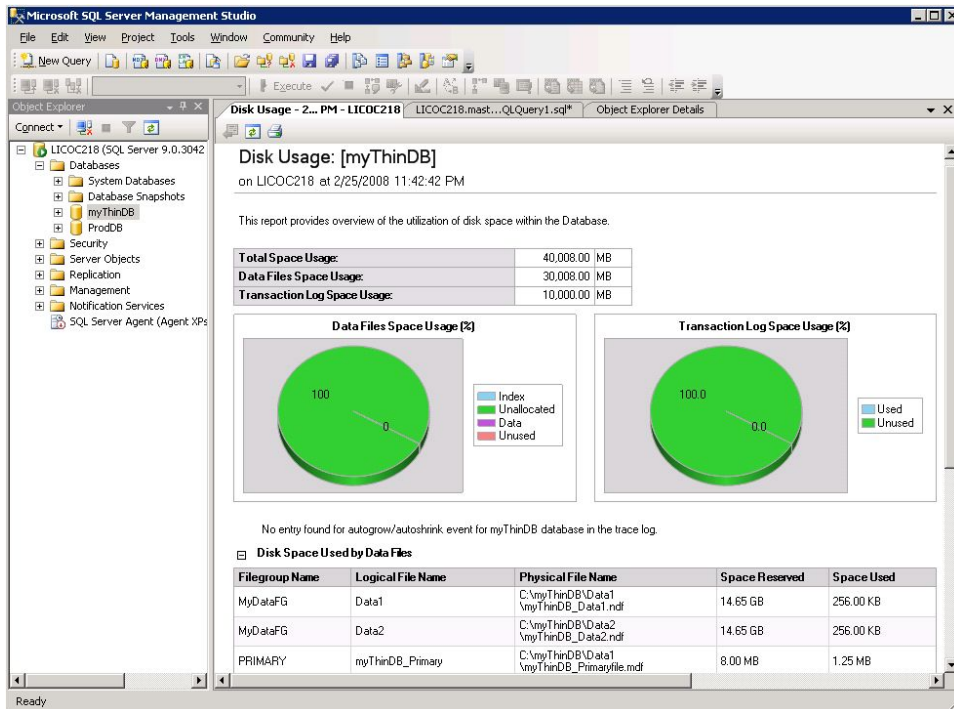


Figure 7. SQL Server Management Studio view of the created database

Finally, when viewing the mount point for the myThinDB_Data1.ndf location (as well as the primary file) it is seen that Windows NTFS considers the data file to be fully allocated. The file space allocation view is based on the size of the file created by Microsoft SQL Server and shows that the file is 15 GB in size.

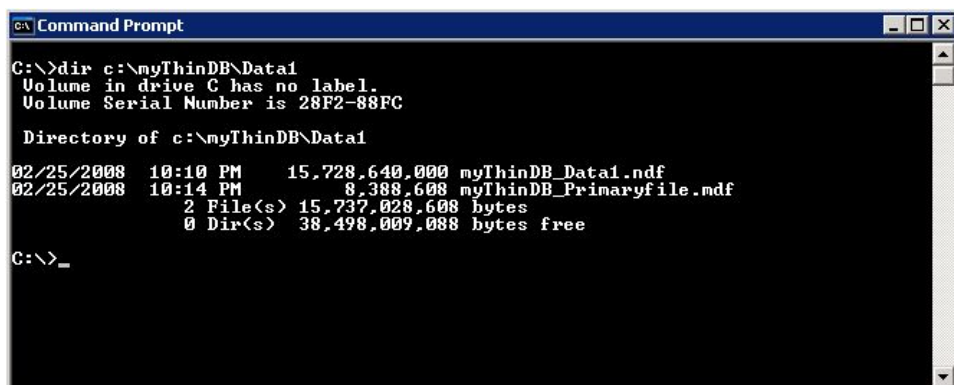


Figure 8. Windows NTFS view of the mount point used for data files

SQL Server TEMPDB database

The usage cycle of Microsoft SQL Server 2005 or 2008 TEMPDB storage deployed on virtually provisioned storage is interesting to follow and understand. Initially, as TEMPDB is simply another SQL Server database the rules for data files and log files apply. The transaction log files will be fully allocated at creation, and data files will be allocated on an as-needed basis.

When temporary space is required from TEMPDB for the first time, thin extents are allocated out of the associated thin pools. However, when the temporary space is released by the database, the thin extents will still remain allocated. This behavior will be consistent across restarts of the Microsoft SQL Server instance,

as extent allocations from thin pools will persist for the life of the thin devices. The fact that TEMPDB may be reinitialized when the SQL Server Instance is restarted will not affect thin extent allocations. The total thin extent allocation will be based on the maximum TEMPDB utilization, but will never exceed the maximum size of the TEMPDB itself (assuming maximum parameters have been set for data files and the transaction log).

Using Microsoft SQL Server Auto-Grow and Virtual Provisioning

Microsoft SQL Server 2005 and 2008 support the concept of data file and transaction log auto-growth. This functionality is defined at the time of creation of the database, or may be modified during the life of the database by using the ALTER DATABASE T-SQL statement.

In general, the Auto-Grow feature of SQL Server is independent of the implementation of Virtual Provisioning. Auto-Grow functionality will behave in exactly the same manner in a thinly provisioned environment as it does in a traditional fully allocated environment.

Microsoft SQL Server and EMC guidance for Auto-Grow is to avoid depending on this functionality, especially in larger environments, where proactive administrator management of file growth is highly recommended. While Auto-Grow functionality will help databases avoid an out-of-space condition if there is available space in the volume where a data file is located, it does not grow all data files of a file group at the same time. Microsoft SQL Server uses a proportional fill mechanism for allocating space for a table or index across a given file group. If one of the data files executes an Auto-Grow operation, it will be seen to have proportionally more free space, and will therefore have more table extent allocations made from it. This in turn may skew workloads.

To ensure even distribution of allocations and workload patterns in more complex database environments that implement multiple data files within a file group, it is highly recommended to manually extend all the data files within a file group at the same time. This will ensure that the proportional fill mechanism continues to evenly distribute data across all data files.

Space reclamation

Solutions Enabler 7.1, in conjunction with the Enginuity 5874 Q4 service release, now provides the ability to free up, also referred to as “de-allocate,” storage ranges found to contain all zeros. Space reclamation was designed to be used primarily following a replication or migration activity from a regular device to a thin device, in which software tools such as Open Replicator and Open Migrator (for certain environments) copy all-zero, unused space to the target thin volume. This feature is an extension of the existing Virtual Provisioning space de-allocation mechanism. Previous versions of Enginuity and Solutions Enabler allowed the reclamation of allocated but unused thin device space from a thin pool. Administrators now have the ability to reclaim all-zero space, including both host-unwritten extents that contain all zeros due to file system or database formatting methods. The space reclamation process is nondisruptive and can be executed with the targeted thin device online to operating systems and applications.

Starting the space reclamation process will spawn a back-end disk director (DA) task that will examine the allocated thin device extents on specified thin devices. A thin device extent is 768 KB (or 12 tracks) in size and is the default unit of storage at which data is allocated for a thin device. For each allocated extent, all 12 tracks will be brought into Symmetrix cache and examined to see if they contain all zero data. If the entire extent contains all zero data, the extent will be de-allocated and added back into the pool, making it available for a new extent allocation operation. Should any portion of the extent contain non-zero data, none of the extent will be reclaimed.

Space reclamation considerations for SQL Server

Transaction log files

Active log files are formatted when a Microsoft SQL Server database is created. Every page of the log files is written to at the time of creation so that the log files become fully provisioned when they are initialized. Because the log file will mostly contain contiguous zeros when first written, its storage has the potential to be reclaimed by Virtual Provisioning reclamation. Over time, as transactions are written to the log files and later truncated based on log backup or checkpoint operations, the log file will no longer contain all zero data. With this behavior in mind, it should not be expected that a log file allocated to a proper size and used for a given period of time will contain contiguous zeros sufficient enough to be reclaimed.

Should a transaction log file be sufficiently oversized to where areas of the log were never used by SQL Server, there is the potential that these areas on disk could be reclaimed. While it is possible to reclaim unused portions of the SQL log file it is generally not recommended to reclaim zero space dedicated to a log device. Pre-written space on thin devices provides optimum write performance, important for the SQL log file. Additionally, pre-written space will guarantee thin pool allocations are dedicated and available for the SQL log file.

Database files

Prior to SQL Server 2005, SQL Server would fully initialize all datafile and transaction log file components, writing to every page in all database files and transaction log files. The process of writing to every page during initialization writes contiguous zeros to areas of the database not otherwise used by file metadata pages. If these zero pages remain untouched by SQL Server there is the potential that the areas on disk could be reclaimed.

As table and index information is written to a fully initialized datafile, areas of the database become allocated and used by non-zero user data. Assuming a database file was overallocated to where its initialized size is larger than the amount of user data in tables and indexes it may be possible to reclaim the remaining zero space, assuming it was never touched by SQL Server outside of the initialization process during creation or growth operations. Figure 9 provides a logical depiction of a file system and database file in this state.

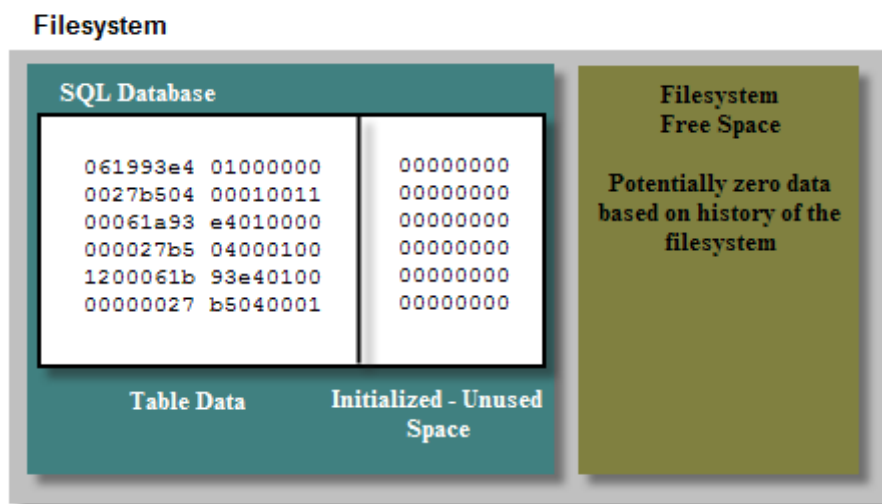


Figure 9. File system and SQL Server database file with initialized and unused space

As previously discussed, Microsoft SQL Server 2005 introduced new functionality that altered the behavior during the database creation phase. As of SQL Server 2005, the database creation phase was no longer required to initialize all datafiles if Instant File Initialization could be utilized by the SQL Server environment.

Transaction log files will continue to be fully allocated with zeros even when using Instant File Initialization.

Instant File Initialization will be used automatically by SQL Server provided the service account, under which the SQL Server service is running, has the “Perform volume maintenance tasks” user rights assignment under the local security policy. By default, only users marked as Administrators have this permission.

Areas of a disk under which a sparse file is defined, as created by Instant File Initialization, are not zeroed. Whatever data existed on these areas of the disk, be it zeros or perhaps non-zeros based on the history of the file system and LUN, will remain. In the context of migrations, should a LUN-based copy be replicated from a source volume where a database was created with Instant File Initialization, to a target thin device, the ability to reclaim zero written data will depend on the history of the LUN and file system. Figure 10 depicts this scenario.

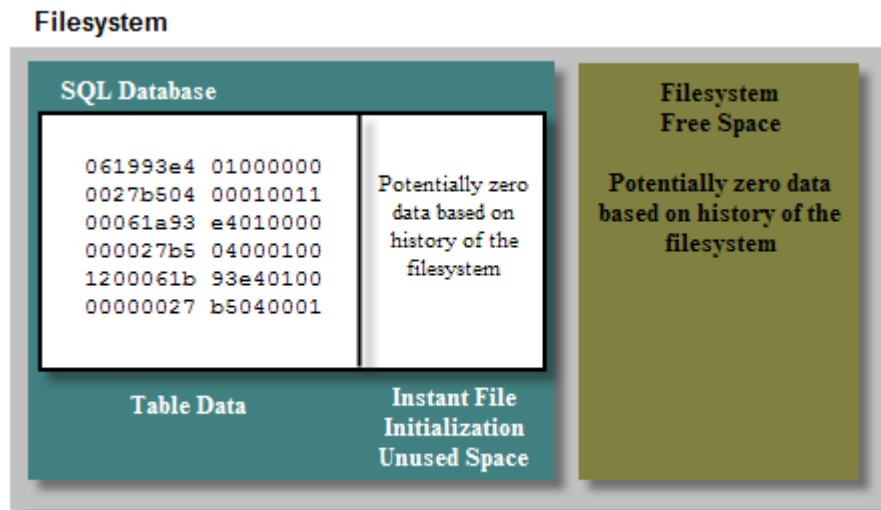


Figure 10. File system and SQL Server database created with Instant File Initialization

Should user data have ever resided in areas of the database file now considered free based on row or table delete operations, those areas will be non-zero and will not have the potential to be reclaimed. With SQL Server 2008 there is a stored procedure called `sp_clean_db_free_space`, which is intended to clean areas of a database file that are considered free. This procedure is also included with SQL Server 2005 SP3 but is undocumented. The `sp_clean_db_free_space` command will zero not only previously used areas of the database file, but also areas of the file that have never been used. This operation will subsequently cause writes and thin pool allocations on areas of the database file that have not been allocated where Instant File Initialization was used. As a part of the cleaning process each page touched by the procedure will leave behind database header information. While the majority of the page is zeroed, the page header left behind makes it impossible to reclaim the space cleaned by this utility. Because of this behavior, it is not recommended to use `sp_clean_db_free_space` as a means to reclaim space within a SQL database file.

For more application considerations regarding space reclamation please see the white paper *EMC Symmetrix V-Max Virtual Provisioning Space Reclamation and Application Considerations*, available on Powerlink.

Local and remote replication between thin devices

Organizations will be able to perform “thin to thin” replication with Symmetrix thin devices by using standard TimeFinder®, SRDF®, and Open Replicator operations. This includes TimeFinder/Snap, TimeFinder/Clone, SRDF/Synchronous, SRDF/Asynchronous, and SRDF/Data Mobility.

In addition, thin devices can be used as control devices for hot and cold pull and cold push Open Replicator copy operations. If a push operation is done using a thin device as the source, zeroes will be sent for any regions of the thin device that have not been allocated, or that have been allocated but have not been written to.

Open Replicator can also be used to copy data from a regular device to a thin device. If a pull or push operation is initiated from a regular device that targets a thin device, then a portion of the target thin device, equal in size to the reported size of the source volume, will become allocated.

Local replication between thin and regular devices

TimeFinder/Clone supports the creation of clone sessions between regular devices and thin devices on Symmetrix V-Max arrays as of the Engenuity 5874 Q4 2009 service release.

When a clone copy is made between a regular source device and a thin target device, thin device extents that have never been written to by a host will not be copied to the target volume. Following the clone copy, any thin device extents that were allocated on the clone target that contain all zeros can be reclaimed and added back to the target thin device’s pool. This is accomplished using Virtual Provisioning space reclamation.

Performance considerations

The architecture of Virtual Provisioning creates a naturally striped environment where the thin extents are allocated across all volumes in the assigned storage pool. The larger the storage pool for the allocations is, the greater number of devices that can be leveraged for the Microsoft SQL Server database I/O.

One of the possible consequences of using a large pool of data devices and potentially sharing these devices with other applications is variability in performance. In other words, possible contention with other applications for physical disk resources may cause inconsistent performance levels. If this variability is not desirable for a particular application, that application could be dedicated to its own thin pool. The Symmetrix supports up to 512 thin pools.

When a new data extent is required from the thin pool there is an additional latency introduced to the write while the thin extent location is assigned and formatted. This latency is approximately 1 millisecond, and is incurred only when a new thin extent allocation is required. Thus for a sequential write stream, a new extent allocation will occur on the first new allocation, and again when the current stripe has been fully written and the writes are moved to a new extent. If the application cannot tolerate this occasional additional latency it is recommend to preallocate storage to the thin device when the thin device is bound to the thin pool.

Automated pool rebalancing

Starting in the Engenuity 5874 Q4 2009 service release and Solutions Enabler 7.1 with Symmetrix V-Max arrays, automated pool rebalancing allows the user to run a balancing operation that will redistribute data evenly across the enabled data devices in the thin pool. Because the thin extents are

allocated from the thin pool in round-robin fashion, the rebalancing mechanism will be used primarily when adding data devices to increase thin pool capacity.

The balancing algorithm will calculate the minimum, maximum, and mean used capacity values of the data devices in the thin pool. The Symmetrix will then move thin device extents from the data devices with the highest used capacity to those with the lowest until the pool is balanced.

When a thin pool is grown and subsequently rebalanced, depending on the physical location of the data devices added to the pool, there is the potential for improved performance. For example, should an existing pool that has data devices residing on 24 physical disks be grown and rebalanced onto 48 physical disks, not only will capacity be increased, but existing applications within the pool may see a performance gain.

To show the potential performance benefits of automated pool rebalancing, a test was run with an 888 GB SQL Server 2008 database being used for OLTP activity. The existing thin pool supporting this database was comprised of 12 mirrored data devices that resided on 24 physical spindles, with the database residing on 12 thin devices bound to the pool as shown here:

```
C:\>symcfg show -thin -pool sqlbalance -detail -sid 67
```

```
Symmetrix ID           : 000192601667
Pool Name               : sqlbalance
Pool Type              : Thin
Dev Emulation          : FBA
Dev Configuration      : 2-Way Mir
Pool State             : Enabled
# of Devices in Pool   : 12
# of Enabled Devices in Pool : 12
Max. Subscription Percent : None
```

Enabled Devices(12):

```
{
  -----
  Sym      Total      Alloc      Free Full  Device
  Dev      Tracks     Tracks     Tracks (%)  State
  -----
  01D1      1965588    1054716    910872   53   Enabled
  01D2      1965588    1053624    911964   53   Enabled
  01D3      1965588    1052292    913296   53   Enabled
  01D4      1965588    1053372    912216   53   Enabled
  01D5      1965588    1052544    913044   53   Enabled
  01D6      1965588    1054008    911580   53   Enabled
  01D7      1965588    1055376    910212   53   Enabled
  01D8      1965588    1053972    911616   53   Enabled
  01D9      1965588    1053372    912216   53   Enabled
  01DA      1965588    1051176    914412   53   Enabled
  01DB      1965588    1053276    912312   53   Enabled
  01DC      1965588    1052088    913500   53   Enabled
  -----
  Tracks    23587056    12639816    10947240   53
}
```

Thin Devices(12):

```
{
  -----
  -----
}
```

Sym Dev Status	Total Tracks	Pool Subs (%)	Pool Allocated Tracks (%)	Pool Written Tracks (%)			
01C5	1965600	8	1326492	6	1284080	6	Bound
01C6	1965600	8	1513116	6	1472292	6	Bound
01C7	1965600	8	764844	4	753310	4	Bound
01C8	1965600	8	1921320	8	1921237	8	Bound
01C9	1965600	8	1509024	6	1468598	6	Bound
01CA	1965600	8	1510284	6	1469629	6	Bound
01CB	1965600	8	1796256	8	1794997	8	Bound
01CC	1965600	8	1508148	6	1467650	6	Bound
01CD	1965600	8	764868	4	753308	4	Bound
01CE	1965600	8	9456	0	9380	0	Bound
01CF	1965600	8	8016	0	7678	0	Bound
01D0	1965600	8	7992	0	7656	0	Bound
Tracks	23587200	100	12639816	54	12409815	53	

Database performance in the above configuration, given the high number of database IOPS (shown in Figure 11), was showing degraded performance. Specifically the read performance against the thin pool was at an average of over 40 ms response time for many of the files supporting the database, as shown in Figure 12.



Figure 11. Database file IOPS before pool growth and balancing

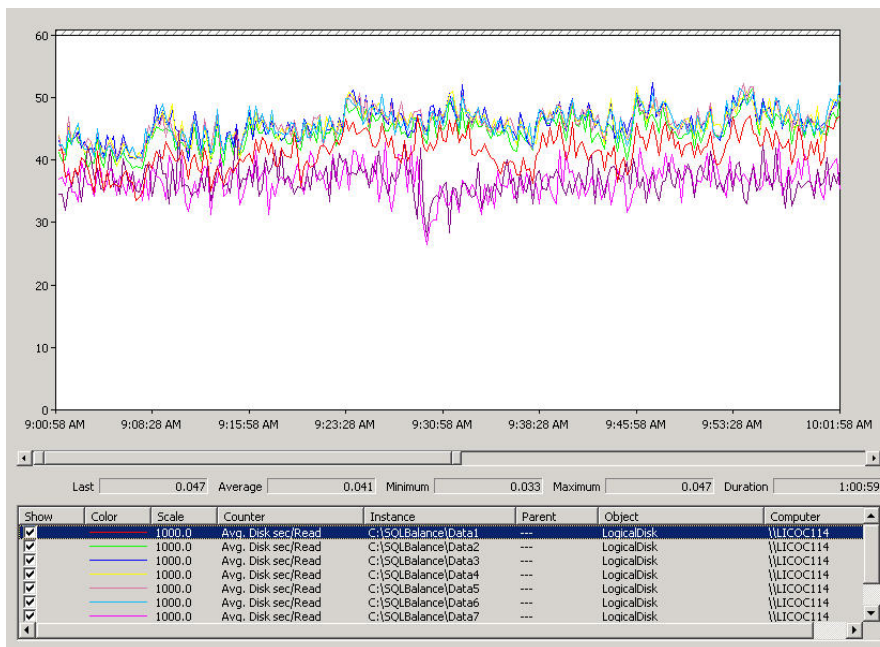


Figure 12. Database file read response time before pool growth and balancing

To help improve performance, an additional 12 mirrored data devices residing on an additional 24 physical drives were added and subsequently used to rebalance the pool. Once added to the pool using SMC or the symconfigure command the thin pool looked as follows:

```
C:\>symcfg show -thin -pool sqlbalance -detail -sid 67
```

```
Symmetrix ID: 000192601667
```

```
Symmetrix ID           : 000192601667
Pool Name              : sqlbalance
Pool Type              : Thin
Dev Emulation          : FBA
Dev Configuration     : 2-Way Mir
Pool State             : Enabled
# of Devices in Pool   : 24
# of Enabled Devices in Pool : 24
Max. Subscription Percent : None
```

```
Enabled Devices(24) :
```

```
{
```

Sym	Total	Alloc	Free	Full	Device
Dev	Tracks	Tracks	Tracks	(%)	State
01D1	1965588	1054716	910872	53	Enabled
01D2	1965588	1053624	911964	53	Enabled
01D3	1965588	1052292	913296	53	Enabled
01D4	1965588	1053372	912216	53	Enabled
01D5	1965588	1052544	913044	53	Enabled
01D6	1965588	1054008	911580	53	Enabled
01D7	1965588	1055376	910212	53	Enabled

```

01D8      1965588      1053972      911616      53      Enabled
01D9      1965588      1053372      912216      53      Enabled
01DA      1965588      1051176      914412      53      Enabled
01DB      1965588      1053276      912312      53      Enabled
01DC      1965588      1052088      913500      53      Enabled
01DD      1965588           12      1965576      0      Enabled
01DE      1965588           12      1965576      0      Enabled
01DF      1965588           12      1965576      0      Enabled
01E0      1965588           12      1965576      0      Enabled
01E1      1965588           12      1965576      0      Enabled
01E2      1965588           12      1965576      0      Enabled
01E3      1965588           12      1965576      0      Enabled
01E4      1965588           12      1965576      0      Enabled
01E5      1965588           12      1965576      0      Enabled
01E6      1965588           12      1965576      0      Enabled
01E7      1965588           12      1965576      0      Enabled
01E8      1965588           12      1965576      0      Enabled
-----
Tracks    47174112      12639960      34534152      26
}

Thin Devices(12):
{
-----
-----
Sym      Total      Pool      Pool      Pool
Dev      Tracks    Subs    Allocated    Written
Status                                Tracks    (%)    Tracks    (%)    Tracks    (%)
-----
-----
01C5      1965600      4      1326492      3      1284080      3      Bound
01C6      1965600      4      1513116      3      1472292      3      Bound
01C7      1965600      4      764844      2      753310      2      Bound
01C8      1965600      4      1921464      4      1921381      4      Bound
01C9      1965600      4      1509024      3      1468598      3      Bound
01CA      1965600      4      1510284      3      1469629      3      Bound
01CB      1965600      4      1796256      4      1794997      4      Bound
01CC      1965600      4      1508148      3      1467650      3      Bound
01CD      1965600      4      764868      2      753308      2      Bound
01CE      1965600      4      9456      0      9380      0      Bound
01CF      1965600      4      8016      0      7678      0      Bound
01D0      1965600      4      7992      0      7656      0      Bound
-----
Tracks    23587200      50      12639960      27      12409959      26
}

```

In order to rebalance the pool so that all 48 physical spindles would be utilized for the previously existing database tables and workload, the following command was issued:

```
C:\>symconfigure -cmd "start balancing on pool sqlbalance;" commit -
sid 67
```

Following the rebalance command the pool will enter a “balancing” state where data will be copied to the new devices until all data devices are within at least 10% (or less) difference in the amount of space used.

Host I/O will take priority over the balancing activity, so the amount of time the rebalance will take greatly depends on the amount of data to rebalance, the number of devices available within the pool, and the host workload.

In the above example the final pool state, once the balancing activity was complete, was the following:

```
C:\>symcfg show -thin -pool sqlbalance -detail -sid 67
```

Symmetrix ID: 000192601667

```
Symmetrix ID           : 000192601667
Pool Name              : sqlbalance
Pool Type              : Thin
Dev Emulation          : FBA
Dev Configuration      : 2-Way Mir
Pool State             : Enabled
# of Devices in Pool   : 24
# of Enabled Devices in Pool : 24
Max. Subscription Percent : None
```

Enabled Devices (24) :

```
{
-----
Sym      Total      Alloc      Free Full  Device
Dev      Tracks     Tracks     Tracks  (%)   State
-----
01D1      1965588      580416      1385172    29  Enabled
01D2      1965588      579312      1386276    29  Enabled
01D3      1965588      582504      1383084    29  Enabled
01D4      1965588      579072      1386516    29  Enabled
01D5      1965588      578244      1387344    29  Enabled
01D6      1965588      579720      1385868    29  Enabled
01D7      1965588      581040      1384548    29  Enabled
01D8      1965588      579672      1385916    29  Enabled
01D9      1965588      579072      1386516    29  Enabled
01DA      1965588      576876      1388712    29  Enabled
01DB      1965588      578976      1386612    29  Enabled
01DC      1965588      577788      1387800    29  Enabled
01DD      1965588      474948      1490640    24  Enabled
01DE      1965588      474948      1490640    24  Enabled
01DF      1965588      474960      1490628    24  Enabled
01E0      1965588      474972      1490616    24  Enabled
01E1      1965588      474972      1490616    24  Enabled
01E2      1965588      474972      1490616    24  Enabled
01E3      1965588      474972      1490616    24  Enabled
01E4      1965588      471792      1493796    24  Enabled
01E5      1965588      473628      1491960    24  Enabled
01E6      1965588      474972      1490616    24  Enabled
01E7      1965588      474972      1490616    24  Enabled
01E8      1965588      474972      1490616    24  Enabled
-----
Tracks    47174112    12647772    34526340    26
}
```

Thin Devices(12):

Sym	Total	Pool	Pool		Pool		
Dev	Tracks	Subs	Allocated		Written		
Status		(%)	Tracks	(%)	Tracks	(%)	

01C5	1965600	4	1327344	3	1284942	3	Bound
01C6	1965600	4	1513968	3	1473156	3	Bound
01C7	1965600	4	765324	2	753800	2	Bound
01C8	1965600	4	1921464	4	1921381	4	Bound
01C9	1965600	4	1509876	3	1469460	3	Bound
01CA	1965600	4	1511136	3	1470494	3	Bound
01CB	1965600	4	1796256	4	1794997	4	Bound
01CC	1965600	4	1509000	3	1468513	3	Bound
01CD	1965600	4	765360	2	753801	2	Bound
01CE	1965600	4	9456	0	9380	0	Bound
01CF	1965600	4	9300	0	8869	0	Bound
01D0	1965600	4	9288	0	8841	0	Bound

Tracks	23587200	50	12647772	27	12417634	26	

With the thin pool balanced, the performance of the thin devices supporting the database showed a marked improvement. With the same workload as before, the read response time dropped from 40 ms averages to 3 to 5 ms averages, as shown in Figure 13.

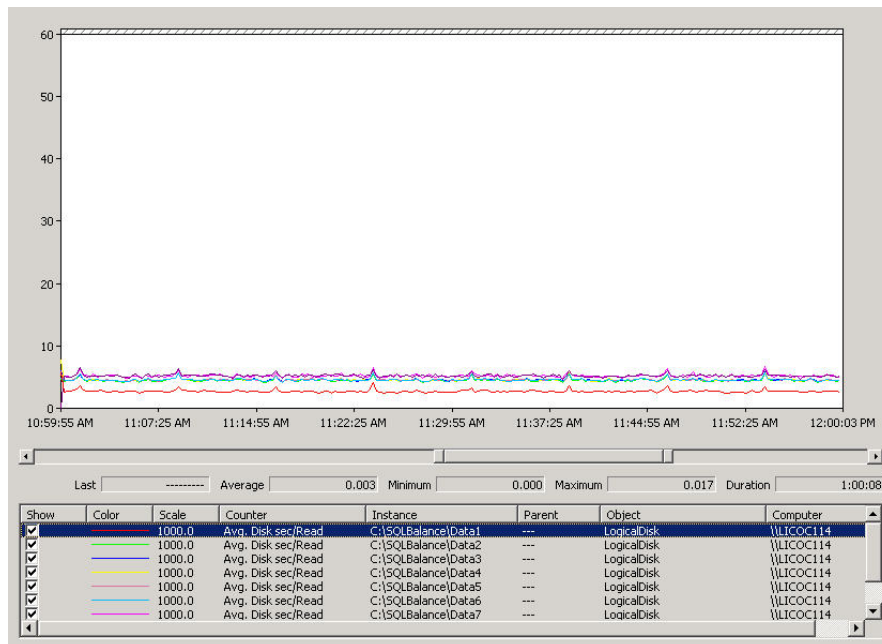


Figure 13. Database file read response time after pool growth and balancing

With the significant drop in read response time, the overall IOPS the application could perform increased for all files in the database, as shown in Figure 14.



Figure 14. Database file IOPS after pool growth and balancing

Thin pool management

When storage is provisioned from a thin pool to support multiple thin devices there is usually more “virtual” storage provisioned to hosts than is supported by the underlying data devices. This is one of the main reasons for using Virtual Provisioning. However, there is a possibility that applications using a thin pool may grow rapidly and request more storage capacity from the thin pool than is actually there. This condition is known as oversubscription. This is an undesirable situation. The next section discusses the steps necessary to avoid oversubscription.

Thin pool monitoring

Along with Virtual Provisioning come several methodologies to monitor the capacity consumption of the thin pools. The Solutions Enabler 6.5 `symcfg monitor` command can be used to monitor pool utilization, as well as displaying the current allocations through the `symcfg show pool` option. There are also event thresholds that can be monitored through the SYMAPI event daemon, thresholds that can be set with the Symmetrix Management Console, and SNMP traps that can be sent for monitoring by EMC ControlCenter[®] or any data center management product, including Microsoft System Center Operations Manager.

As thresholds for thin pool storage are met or exceeded, events are raised by the storage array. In a configuration where the SYMAPI event daemon is implemented on any system with appropriate connectivity to the Symmetrix DMXT[™] array, the event daemon will propagate events into the configured location. This location can include one or more of the following: the Windows event log, an SNMP target, or a defined log file. When configured to log events in to the Windows application event log, events of the form shown in Figure 15 will be posted when threshold values are met or exceeded.

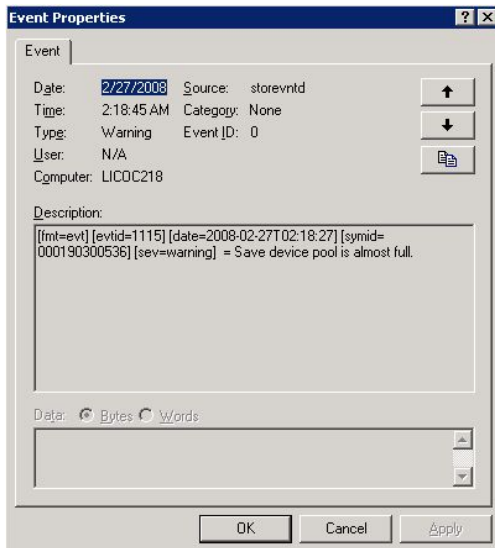


Figure 15. Windows event log entry for the SYMAPI event daemon

System administrators and storage administrators must put processes in place to monitor the capacity for thin pools to make sure that they do not get filled. The pools can be dynamically expanded to include more data devices without application impact. These devices should be added in groups and should be added long before the thin pool is approaching a full condition. If devices are added individually, hot spots on disks can be created when much of the write activity is suddenly directed to the newly added data devices because other data devices in the pool are full. Concerns around creating hot spots can be mitigated with the use of Automated Pool Rebalancing as detailed in the “Performance considerations” section. The addition of new data devices to a thin pool is executed in the same manner as the initial allocation of data devices, and is detailed in the section “Appendix: Virtual Provisioning examples.”

Exhaustion of oversubscribed pools

If a thin pool is oversubscribed and has no available space for new extent allocations, attempts by SQL Server to write to a page that would require a new data extent to be allocated will result in a continual redriving of the I/O operation. Informational messages of the form shown in Figure 16 will be posted to the application event log.

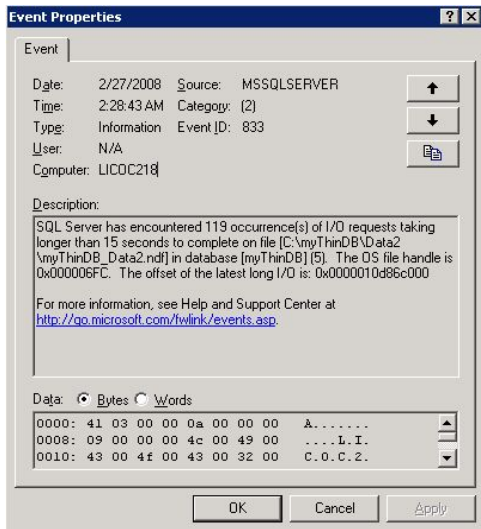


Figure 16. SQL Server 2005 I/O request informational message

SQL Server will continue to buffer changed pages in memory until such time as SQL Server memory is completely filled, or the outstanding writes are able to be serviced. Storage administrators should take immediate action in the event that these errors are logged as a result of exhaustion of thin pool storage. In concert with the I/O delay message, the Windows System log may also report disk errors of event ID 11. These messages are a result of write requests to the thin device being rejected by the Symmetrix target, as new allocations are not possible as the pool is full.

In the event that additional data devices are added and enabled within the thin pool, Microsoft SQL Server will write all outstanding data pages to the data files, and will return to normal operating mode for the database. The resubmission of the outstanding write requests will be successful at the time additional storage space is available to the thin pool. Subsequently all informational messages from SQL Server and disk event ID 11 messages will cease to be generated.

The database is protected at all times from any inconsistencies by the Write Ahead Logging (WAL) mechanism used by Microsoft SQL Server. As stated in the “Transaction log files” section, the transaction log will be fully allocated from any thin pool it may be bound to. As a result, the transaction log itself will not suffer from an overallocation issue. At all times, transaction log records will be written to the transaction log, and will ensure that transactional consistency is maintained even in the event of an abnormal termination of the server.

Conclusion

EMC Symmetrix Virtual Provisioning provides a simple, noninvasive, and economical way to provide storage for Microsoft SQL Server databases. Microsoft Windows Server 2003 or Windows Server 2008 in concert with Microsoft SQL Server 2005 or 2008 in particular provides the specific features required to obtain the maximum benefits of Virtual Provisioning. With Microsoft Windows “thin-friendly” NTFS formatting capabilities, and Microsoft SQL Server 2005 or 2008 Instant File Initialization mechanisms, customers are able to derive the maximum benefits of Virtual Provisioning.

The following summarizes the optimal configurations of Virtual Provisioning and Microsoft SQL Server:

- Microsoft Windows Server 2003 or Windows Server 2008 with NTFS format storage allocation
- Microsoft SQL Server configurations with Instant File Initialization
- Systems where overallocation of storage is typical
- Systems where rapid growth is expected over time but downtime is limited

The following are situations where Virtual Provisioning may not be optimal:

- Systems where shared allocations from a common pool of thin devices is not desired
- Systems where data is deleted and re-created rapidly and the underlying data management structure does not allow for reuse of the deleted space
- Systems that cannot tolerate an occasional response time increase of approximately 1 millisecond, due to writes to uninitialized blocks

References

The following can be found on EMC.com and Powerlink:

White papers

- *New Features in EMC Enginuity 5773 for Symmetrix Open Systems Environments*
- *New Features in EMC Enginuity 5874 for Symmetrix Open Systems Environments*
- *EMC Enginuity 5874 Enhancements for Symmetrix V-Max in Open Systems Environments*
- *EMC Symmetrix V-Max Virtual Provisioning Space Reclamation and Application Considerations*
- *Using diskpar and diskpart to Align Partitions on Windows Basic and Dynamic Disks*

Technical note

- *Best Practices for Fast, Simple Capacity Allocation with EMC Symmetrix Virtual Provisioning*

Feature sheet

- *Symmetrix Virtual Provisioning Feature Specification*

Customer manuals

- *EMC Solutions Enabler Symmetrix Array Management CLI Product Guide*
- *EMC Solutions Enabler Symmetrix Array Controls CLI Product Guide*
- *EMC Solutions Enabler Symmetrix CLI Command Reference*

The following product documentation can be found on Microsoft's website:

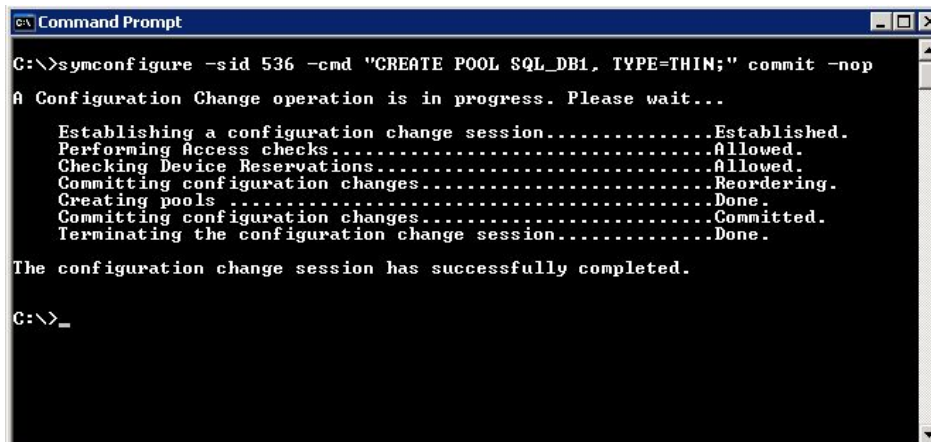
- *Microsoft SQL Server 2005 Books Online*
- *Microsoft SQL Server 2008 Books Online*

Appendix: Virtual Provisioning examples

The following examples provide further details of commands used to create the example environment documented in this white paper. They are provided as examples only, and may not represent EMC best practice recommendations.

Create a thin pool

A thin pool initially needs to be created, such that data devices may be added to provide the storage, and thin devices may be bound to the pool. Figure 17 shows the creation of a pool named “sql_db1” using the symconfigure command.

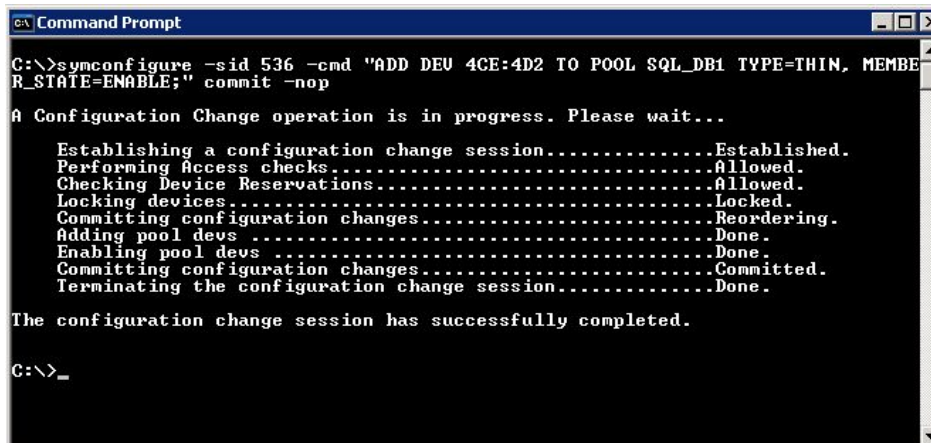


```
C:\>symconfigure -sid 536 -cmd "CREATE POOL SQL_DB1, TYPE=THIN;" commit -nop
A Configuration Change operation is in progress. Please wait...
Establishing a configuration change session.....Established.
Performing Access checks.....Allowed.
Checking Device Reservations.....Allowed.
Committing configuration changes.....Reordering.
Creating pools .....Done.
Committing configuration changes.....Committed.
Terminating the configuration change session.....Done.
The configuration change session has successfully completed.
C:\>_
```

Figure 17. Creation of a thin pool

Add data devices to a thin pool

A thin pool is required to contain enabled data devices from which storage allocations can be made for the bound thin devices. To add devices to a given pool, and set the data device to be enabled, so as to be able to make allocations against the data device, a symconfigure command as shown in Figure 18 may be executed. In the example, a range of devices is being processed. Equally, individual data devices may be added in a similar process.



```
C:\>symconfigure -sid 536 -cmd "ADD DEV 4CE:4D2 TO POOL SQL_DB1 TYPE=THIN, MEMBE
R_STATE=ENABLE;" commit -nop
A Configuration Change operation is in progress. Please wait...
Establishing a configuration change session.....Established.
Performing Access checks.....Allowed.
Checking Device Reservations.....Allowed.
Locking devices.....Locked.
Committing configuration changes.....Reordering.
Adding pool devs .....Done.
Enabling pool devs .....Done.
Committing configuration changes.....Committed.
Terminating the configuration change session.....Done.
The configuration change session has successfully completed.
C:\>_
```

Figure 18. Adding data devices to a thin pool

To display the currently allocated data devices within the pool, and to view their status (whether enabled for allocations is displayed), a command of the form shown in Figure 19 may be used.

```

c:\ Command Prompt
C:\>symcfg -sid 536 show -pool SQL_DB1 -thin -detail

Symmetrix ID: 000190300536

Symmetrix ID      : 000190300536
Pool Name         : SQL_DB1
Pool Type         : Thin
Dev Emulation     : FBA
Dev Configuration : 2-Way Mir
Pool State        : Enabled
# of Devices in Pool : 5
# of Enabled Devices in Pool : 5
Max. Subscription Percent : None

Enabled Devices(5):
<
-----
Sym   Total   Alloc   Free   Full   Device
Dev   Tracks  Tracks  Tracks  (%)   State
-----
04CE   276204      0   276204      0  Enabled
04CF   276204      0   276204      0  Enabled
04D0   276204      0   276204      0  Enabled
04D1   276204      0   276204      0  Enabled
04D2   276204      0   276204      0  Enabled
-----
Tracks 1381020      0 1381020      0
>

No Thin Devices Bound to Device Pool SQL_DB1

C:\>_

```

Figure 19. Display devices configured in a thin pool

Bind thin devices to a pool

Once data devices have been allocated and enabled for use within a given pool, it is possible to bind thin devices to the thin pool. This will cause some extent allocations to be made from the pool, and will enable the thin device to be accessible to the host. Additionally, necessary mapping, masking, and zoning operations will need to be executed for the host to access the devices – this functionality is beyond the scope of this white paper.

Figure 20 provides an example of binding a thin device to a thin pool previously created and populated with data devices.

```

c:\ Command Prompt
C:\>synconfigure -sid 536 -cmd "BIND TDEV 432 TO POOL SQL_DB1;" commit -nop
A Configuration Change operation is in progress. Please wait...

Establishing a configuration change session.....Established.
Processing symmetrix 000190300536
Performing Access checks.....Allowed.
Checking Device Reservations.....Allowed.
Locking devices.....Locked.
Validating configuration changes.....Validated.
Preparing configuration changes.....Prepared.
Committing configuration changes.....Started.
Binding devices.....Done.
Committing configuration changes.....Committed.
Terminating the configuration change session.....Done.

The configuration change session has successfully completed.

C:\>_

```

Figure 20. Bind thin devices to the thin pool

Once thin devices have been bound to a given thin pool, it is possible to see both the data devices with the group as well as thin devices bound to the thin pool. This is shown in Figure 21 and it can be seen that the thin pool has been overallocated. Currently the thin devices account for 140% allocation of the data devices, and are shown in the final line of the example under the Pool Subs (%) column, which details the subscription level of the thin devices from the current thin pool allocation size.

```

C:\>symcfg -sid 536 show -pool SQL_DB1 -thin -detail

Symmetrix ID: 000190300536

Symmetrix ID       : 000190300536
Pool Name          : SQL_DB1
Pool Type          : Thin
Dev Emulation      : FBA
Dev Configuration  : 2-Way Mir
Pool State         : Enabled
# of Devices in Pool : 5
# of Enabled Devices in Pool : 5
Max. Subscription Percent : None

Enabled Devices(5):
<
-----
Sym   Total   Alloc   Free   Full   Device
Dev   Tracks  Tracks  Tracks  <%>   State
-----
04CE   276204      0   276204    0  Enabled
04CF   276204     24   276180    0  Enabled
04D0   276204     12   276192    0  Enabled
04D1   276204     24   276180    0  Enabled
04D2   276204     24   276180    0  Enabled
-----
Tracks 1381020      84  1380936    0
>

Thin Devices(3):
<
-----
Sym   Total   Pool   Pool   Pool
Dev   Tracks  Subs   Alloc   Writen
              <%>   Tracks  <%>   Tracks  <%>   Status
-----
0432   828630     60     36    0      3    0  Bound
0435   828630     60     36    0      3    0  Bound
0438   276210     20     12    0      1    0  Bound
-----
Tracks 1933470    140     84    0      7    0
>

C:\>

```

Figure 21. Display of a thin pool with data devices and thin devices

It can also be seen that the process of binding a thin device to a pool requires a minimal number of allocations. This is shown in the Pool Allocated Tracks column.