Solution Guide

MICROSOFT SQL SERVER BEST PRACTICES AND DESIGN GUIDELINES FOR EMC STORAGE

EMC XtremIO, EMC VMAX Systems, EMC VNX Family, EMC ScaleIO

EMC Solutions

Abstract

This solution guide identifies best practices and key decision points for planning and deploying Microsoft SQL Server with the EMC VMAX® series storage, EMC XtremIO, EMC ScaleIO software-defined storage, and EMC® VNX® family of unified storage.

December 2015

EMC²

Published December 2015

Microsoft SQL Server Best Practices and Design Guidelines for EMC Storage Solution Guide

Part Number H14621

Contents

Chapter 1 Executive Summary

This chapter presents the following topics:

Executive summary

In the planning and design phases of a Microsoft SQL Server implementation, it is important to understand how the application interacts with the storage platform. It is also critical to know storage-design best practices to avoid problems and achieve high performance.

From a storage design perspective, consider the application architecture and user profile characteristics of Microsoft SQL Server for performance, protection, and growth of the SQL Server database.

This paper can help solution professionals assess and address SQL Server storage requirements for performance, scalability, and availability:

- It is always preferable to collect actual data from the site.
- In the absence of the actual performance data, make a series of reasonable assumptions when designing for a typical environment, then monitor and adjust accordingly.
- Always consider protection requirements when designing a storage system.

Document purpose

This paper presents EMC-recommended best practices for storage design to support Microsoft SQL Server. Guidelines are within the context of deploying SQL Server on the EMC XtremIO® all-flash storage, EMC Symmetrix® VMAX® series, EMC® VNX® family and EMC ScaleIO® software-based storage platform. The paper includes guidelines for deploying SQL Server in both physical and virtualized environments.

Audience

This solution guide is intended for EMC customers, EMC partners, and service personnel who are considering implementing a database environment with Microsoft SQL Server or considering upgrading an earlier version of SQL Server. We assume that the audience is familiar with Microsoft SQL Server, EMC storage products such as XtremIO, VMAX3/VMAX, unified VNX, ScaleIO, as well as VMware vSphere or Microsoft Hyper-V virtual environments.

Terminology

This solution guide includes the following terminology.

Table 1. Terminology

Term	Definition
Buffer Pool Extension (BPE)	A new feature introduced in Microsoft SQL Server 2014 that enables the buffer manager to extend SQL Server's usable memory from physical RAM onto solid state disk drives (SSD) for cost-effective increased performance.
Data At Rest Encryption (DARE)	A controller-based encryption method to encrypt all data written to disk, protecting data access against unauthorized drive removal, lost, or stolen drives.

Term	Definition
FAST™ Cache	Fully Automated Storage Tiering (FAST) Cache is EMC software that enables customers to add various flash drive capacities in order to extend existing cache capacity for better system-wide performance. FAST Cache is now available with increased capacity configurations using the 100 GB flash drive or the 200 GB flash drive. These additional configurations are available only on the VNX storage array.
Fully Automated Storage Tiering for Virtual Pools (FAST VP)	A feature of VNX storage arrays that automates the identification of data volumes for the purpose of allocating or reallocating business application data across different performance and capacity tiers within the storage array.
PCIe	A serial expansion bus standard for connecting a computer to peripheral devices such as memory cards.
RAID	Redundant array of independent disks (RAID) is a method of storing data on multiple disk drives to increase performance and storage capacity and to provide redundancy and fault tolerance.
Service Level Objective (SLO)	The response time target for a specified storage group in the VMAX3 system.
Storage pool	Virtual constructs that enable data to move dynamically across different tiers of storage according to the data's business activity. With VNX and VMAX systems, storage pools are fully automated and self-managing.
Transparent Data Encryption (TDE)	The native SQL Server encryption at rest technology, which encrypts the database data and log files.
Thin LUN	A type of LUN created in storage pool in which the physical space allocated can be less than the user capacity seen by the host server.
Thick LUN	A type of LUN created in storage pool in which the physical space allocated is equal to the user capacity seen by the host server
VMDK	Virtual Machine Disk file format in an ESXi Server.
VHDX	Virtual Hard Disk format in Windows Server 2012 Hyper-V.

We value your feedback!

EMC and the authors of this document welcome your feedback on the solution and the solution documentation. Please contact us at EMC.Solution.Feedback@emc.com with your comments.

Authors: Mark Qin Xu, Vincent Shen, Yunlong Zhang, Anthony O'Grady, Haiyan Huang Fiona O'Neill.

Chapter 2 Microsoft SQL Server Components and Architecture

This chapter presents the following topics:

SQL Server overview

The SQL Server family of products includes the relational database management system (commonly known as SQL Server), SQL Server Reporting Services, SQL Server Integration Services, and SQL Server Analysis Service.

Master Data Services (MDS) is another component of the SQL Server product group. MDS is the Microsoft solution for master data management. Master data management (MDM) describes the efforts made by an organization to discover and define non-transactional lists of data, with the goal of compiling maintainable master lists. An MDM project generally includes an evaluation and restructuring of internal business processes as well as the implementation of MDM technology.

While SQL Server is designed to manage static data (primarily lives on disk), StreamInsight is designed to analyze dynamic data that come in streams. A stream is a sequence of data that has time associated with it. For more information, refer to the StreamInsight blog.

With the release of SQL Server 2016, Microsoft will add a new component to the product, called SQL Server R Services (not to be confused with SSRS). The technology for integration with the Revolution R processing engine has already been released in Azure. SQL Server 2016 will be the first on-premises version including an R processor for integration of advanced analytics in applications.

Microsoft frequently markets SQL Server as the data platform that does not require expensive add-ons for key functionality. It is a full set of services and tools that can be purchased using either of two licensing models, Client Access Licensing (CAL) or per core. No add-ons may be an over simplification, however because each SQL Server version is also available in several editions that have a combination of both overlapping and exclusive features as the version cost increases.

A complicating factor for customers assessing how to license SQL Server is that the availability of features in each version tends to change over time. For example, backup compression was available in enterprise edition for 2008 and was added to the standard edition for the 2008R2 release. Different editions with different costs and features requires customers to carefully analyze needs and licensing options regardless of whether the product uses an add-on model or a tiered edition model.

SQL Server 2012 and previous releases

SQL Server was the first enterprise application released by Microsoft for the Windows NT server operating systems. The product emerged from an earlier collaboration with Ashton Tate and Sybase to create a version of the Sybase RDBMS for the IBM OS2 operating system.

For more information on all SQL Server versions and features, refer to the Microsoft SQL Server webpage on MSDN.

SQL Server 2000 was the first version to include elements of the SQLOS. Before this version, SQL Server relied on Windows to provide thread scheduling and memory management. Windows is primarily a general purpose multi-user OS that used preemptive multitasking that proved unsuitable for a RDBMS. Because the time slice allocated to each running thread was small compared to the time it takes for a typical RDBMS task to complete, SQL Server suffered from poor performance as a result of

excessive context switching. The SQL Server team developed the SQLOS to abstract thread and memory management away from Windows. This greatly improved performance. The relationship between Windows and SQL Server use of memory and CPU is also largely responsible for the general recommendation to dedicate hardware (physical or virtual) to the database system. Because SQL Server allocates thread schedulers and memory reservations at startup time, it often does not interact well with other enterprise applications that expect a more cooperative multi-tasking environment. SQLOS is further discussed in the next section.

Microsoft also focused on the development of Business Intelligence functionality for the 2000 release including Data Transformation Services, the first SQL Server ETL tool, and the online analytical processing (OLAP) analysis services release based on technology acquired from Panorama Software a few years earlier. SQL Server Reporting Services was released with Service Pack 4.

SQL Server 2005 involved a large development effort, a complicated joint release with Visual Studio, and a new .NET framwork version to support the inclusion of the Common Language Runtime (CLR) code processing as an external service to the database engine. This release also included the introduction of the XML data type and Dynamic Management Views (DMVS) for monitoring and diagnosing server state and performance.

SQL Server 2005 Service Pack 1 (SP1) added database mirroring for redundancy and failover capability at the database level.

SQL Server 2008 introduced Resource Governor with the ability to throttle CPU and Memory, as well as Transparent Data Encryption.

SQL Server 2008 also introduced a set of new data types for managing large binary objects, spatial data, and file-based objects with transaction consistency support called Filestream.

SQL Server 2008 R2 added Master Data Services to centrally manage master data entities and hierarchies and Multi-Server Management to centralize multiple SQL Server instances and services.

SQL Server 2012 introduced AlwaysOn technologies to address challenges with database mirroring and gave users more optons to reduce downtime and improve access to secondary copies. Database mirroring relied on the semantics of client connection strings to faciliate redirection to a secondary copy in the event the primary copy was not responding. This made it difficult to extend that model beyond two replicas.

AlwaysOn solved the client redirection issue through integration with Microsoft Failover Clustering. The traditional SQL Server failover clustering technology for instance-level protection was integrated with AlwaysOn. Windows failover clustering was also used to manage a virtual name resource for a collection of database replicas, called an Availablity Group, that did not require shared storage.

A limited version of columnstore indexes were a new feature in SQL Server 2012 that were added to increase query performance. Contained databases were introduced with a limited set of features and a commitment to further enhancements to simplify moving databases between instances.

SQL Server 2014

SQL Server 2014 included major new features and enhancements to earlier capabilities.

In-Memory OLTP

In-Memory OLTP was a new feature in SQL Server 2014 (Hekaton). Traditional tables and execution plans are swapped into and out of memory as needed to support the current workload. In-memory technology enables more consistent performance by ensuring that selected tables and stored procedures are not swapped out of memory. The In-Memory OLTP engine supports high concurrency by using a new optimistic concurrency control mechanism to eliminate locking delays.

With In-Memory OLTP, you define a heavily accessed table as memory-optimized. Memory-optimized tables are fully transactional and durable and are accessed by using Transact-SQL in the same way as disk-based tables. Memory-optimized tables still must persist on disk when the instance is shut down and are loaded into memory when the engine is restarted. Indexes on memory-optimized tables are not persisted. This can affect performance on startup and the amount of disk storage required for memory-optimized objects. There are a number of restrictions related to memory-optimized tables and some code may need to change after implementation.

Buffer Pool Extension

Buffer Pool Extension (BPE), introduced in SQL Server 2014, enables the integration of a nonvolatile random-access-memory extension with the Database Engine buffer pool. With the new Buffer Pool Extensions feature, you can use SSD drives to expand the buffer pool in systems that have exhausted the ability to add additional memory. BPE can increase performance for read-heavy OLTP workloads where there is an indication that more memory would be beneficial. There are limits on the recommended size of the extension and only clean pages can be stored in the extension.

BPE benefits database workloads in the following ways:

- Increased random I/O throughput.
- Reduced I/O latency.
- Increased transaction throughput.
- Improved read performance with a larger buffer pool.
- BPE also provides a caching architecture to take advantage of present and future low-cost flash memory drives.

Enhancements for columnstore indexes

Microsoft introduced the columnstore index in SQL Server 2012. It provided improved performance for data warehousing queries; however, the underlying table could not be updated while the index was in place. This made the index best suited for data warehouse scenarios where the index could be dropped prior to a data load and then re-created. SQL Server 2014 has the features of SQL Server 2012 plus updateable clustered columnstore indexes.

Enhancements for Resource Governor

In previous versions, Resource Governor allows the administrator to limit the CPU and memory utilization for a given workload. SQL Server 2014 extends resource management to include storage I/O.

SQL Server 2016

At the time of writing SQL Server 2016 is still in development. Features to be included in the final release may change at any time up to the release date. The following sections are based on what is currently available from Microsoft.

Always Encrypted

Always Encrypted technology adds protection for sensitive data both in the SQL Server buffer pool and over the wire by encrypting data inside client applications. Always Encrypted is implemented for selected columns and the data is also encrypted on disk. By operating at the application layer, the encryption keys are not available to SQL Server administrators. As a result, Always Encrypted provides a separation between roles.

Those who own the data are allowed to view it; however, those who manage the data, such as on-premises database administrators, cloud database operators, or other highly-privileged users cannot access the encrypted data because they are not authorized users. Always Encrypted enables database users to store sensitive data outside of their direct control, and makes encryption transparent to applications.

Stretch Database

Stretch Database allows users to archive less frequently used data using a remote server in an Azure data center. Stretch Database supports only V12 Azure SQL Server database servers. The Stretch Database wizard creates a SQL Server database server with the Standard service tier and the S3 performance level by default. The size of the remote portion for a database enabled for Stretch is limited only by the Azure SQL Database service tier. There are no plans to support another local SQL Server instance as the remote endpoint with Stretch Database.

Stretch Database also requires no changes for existing queries or client applications. While enabling the Stretch Database feature for a local database, the user can specify whether to begin migration immediately or pause until a later time. A backup of a Stretch-enabled database is a shallow backup that does not include the data migrated to the remote server.

Microsoft plans to release the Stretch Database Advisor as a component of SQL Server 2016 Upgrade Advisor to identify databases and tables suitable for the Stretch Database feature. There are a number of SQL Server features that are currently not

compatible with Stretch Database. Refer to Books Online for SQL Server 2016 for more details.

Polybase integration

Polybase is another technology acquisition by Microsoft. Microsoft has developed TSQL extensions that allow users to create virtual schemas on top of data stored in selected Hadoop cluster implementations and then join that with data stored in traditional relational tables. The query processor can optimize query performance by deciding whether to perform heavy data processing using the Hadoop or SQL Server engines. A prerequisite for this feature requires a Java installation on the database server.

SQL Server R Services

SQL Server R Services provides a platform for writing intelligent applications that uncover new insights and create predictions using data accessible by SQL Server. It allows use of the powerful R language and package portfolio to bring analytics close to where the data exists. SQL Server R Services provides familiar Transact-SQL syntax for interfacing your production applications with calls to the R runtime that retrieve predictions and visuals. SQL Server 2016 Enterprise Edition includes access to the Revolution R scalable libraries to overcome open source R's single threaded in-memory processing performance and scale limitations.

In-Memory OLTP enhancements

The next release of SQL Server will reduce or eliminate a number of limitations and restrictions that exist for memory-optimized tables and indexes with SQL Server 2014. The recommended limit for the size of all in-memory objects is increasing from 256 GB to 2 TB if sufficient memory exists. ALTER TABLE will be supported for memory-optimized tables to add, drop, or alter columns, or to add, drop, or rebuild indexes. The data on disk for SQL Server 2016 memory-optimized tables will be encrypted when TDE is enabled for the database.

For the latest information on these and other SQL Server 2016 features, refer to Books Online for SQL Server 2016 on MSDN.

SQL Server database engine architecture

The SQLOS was introduced in SQL 2000 and CLR intregration occurred in SQL Server 2005. The database engine has been stable for many releases with few fundamental changes. SQL Server 2016 includes the integration of an R language processor as an external process callable from the SQL Server database engine in much the same way that the CLR integration works. Microsoft indicated during the PASS Summit 2015 that the integration of Revolution R was done to facilitate the addition of other language processors in the future.

Figure 1 shows a detailed block diagram of the SQL Server database engine architecture. We do not discuss every component in the diagram but rather focus on the areas that will help you understand the relationship between SQL Server and the storage components that it accesses during operations.

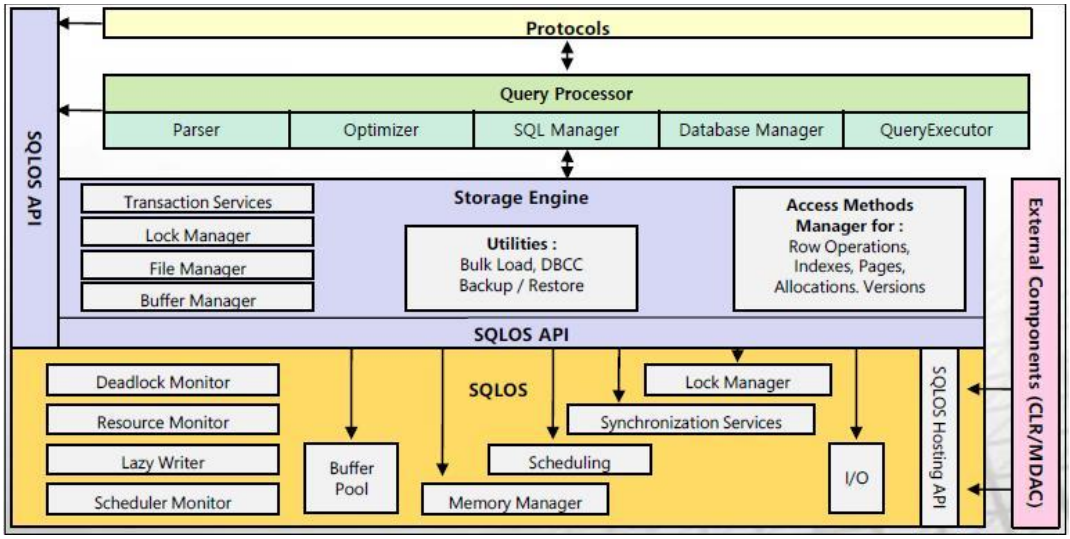

Figure 1. SQL Server architecture

The four major components of the SQL Server architecture are: protocol layer, SQLOS, query processor (relational engine), and storage engine.

Protocol layer

The **protocol layer** manages the communication between clients and the database engine. .NET libraries on both the client and server computers use a Microsoft defined messaging format called Tabular Data Stream (TDS) to exchange requests and results. The protocol layer encapsulates TDS in a standard communication protocal such as TCP/IP or Named Pipes. When the TDS messages that orginate from the client are unpacked from the communication protocol, they are handed off to a command processor that is part of the relational engine. Results sets sent back to the client as TDS are unpacked and processed by the client application.

SQLOS

SQLOS is an abstraction layer that lies between the SQL Server database engine, Windows and any external components such as the CLR and the R language processor. It is responsible for functions including thread scheduling and memory management typically performed by Windows for other applications. SQLOS does not have any abstration for I/O requests. Threads used by SQL Server issue mostly

asynchronous read and write requests directly to Windows and then wait for the results.

The SQLOS queries Windows at startup to detemine the number and type of CPUs (NUMA, non-NUMA). It uses this information to create theads and schedulers to handle multiple simultaneous requests while it hides the details from Windows to prevent unnecessary context switching for long running processes.

The database engine constantly communicates with Windows through the SQLOS to request and respond to memory allocation adjustment requests. Windows is responsible for negotiating memory allocations for all processes running on the server and makes adjustments based on requrests from applications as well as its own needs and the total amount of memory available to Windows. Windows grants memory requests from SQL Server as long as there are sufficient available memory resources on the server. If Windows receives more requests for memory than it can grant, the OS may try to negotiate with SQL Server to reduce the amount of memory allocated.

The SQLOS manages its memory resources dynamically in negotiation with Windows using an internal object called the buffer pool. All memory in the buffer pool that is not used by an internal process such as the procedure cache or client connections, and so on, is allocated to a data cache used by the storage engine for buffering data and index pages. The data cache is typically the largest consumer of memory from the buffer pool. The SQLOS uses a memory broker to efficiently adjust memory allocations from the buffer pool to the many internal services that need dynamic memory allocations.

Query processor

The **Query Processor**, shown in Figure 1, is also referred to as the **relational engine**.

The main responsibilites of the **relational engine** are:

- Validating T-SQL statements.
- Parsing SQL statements by breaking them down into keywords, parameters, operators, and identifiers, and creating a series of smaller logical operations.
- Optimizing the execution plan, which consists of finding an acceptable plan from the list of candidate plans that it determines can perform the tasks required. The relational engine estimates the cost of the processing steps based on internal metrics including estimated memory usage, CPU utilization, and the number of required I/Os based on statistics for a set of competing plans until further optimziation is determined to be more expensive than execution. The optimizer does not guarantee that the selected plan is the best but is good enough to indicate that further optimization is not warranted. Plans that are used from cache and plans that are considered trival require optimization.
- Processing Data Definition Language (DDL) and other statements, such as SET statements, to set connection options and the CREATE statements to create objects in a database.

- Formatting results returned to the client. The results are formatted as either a traditional, tabular result set or as an XML document. The results are then encapsulated in one or more TDS packets and returned to the application.

Storage engine

The SQL Server **storage engine** interacts with the relational engine to provide services to end users. From the perspective of the user and the DBA, the functioning of the storage and relational engines are indistinguishable. However, for IT professionals who design and manage applications, a basic understanding of these internals can be instrumental in understanding SQL Server behavior and problem troubleshooting.

The main functions of the storage engine are:

- Managing the data cache buffers and I/O to the physical files
- Controlling concurrency, managing transactions, locking, and logging
- Managing the files and physical pages used to store data
- Recovering from system faults

The relational engine decides which data satisfies a request and the storage engine makes the data available. The storage engine is also responsible for maintaing data integrity to prevent simulaneous requests from interfering with each other.

This high level time line shows how the relational engine and the storage engine work together to satisfiy a request:

1. Data access activity begins with a query, whether it originates from a user interface or from an automated task. The data request is passed from the protocal stack into the relational engine.
2. The relational engine compiles and optimizes the request into an execution plan. The plan consists of a series of steps that is required to retrieve and process the data into a result that the client can consume.
3. The relational engine runs the execution plan. The execution steps that involve accessing tables and indexes are sent to the storage engine.
4. The storage engine returns data to the relational engine where it is combined and processed into the final result set and returned to the protcol stack.
5. The result set is sent back to the user via the protocol stack.

SQL Server logical components

The SQL Server database engine was originally designed to effectively support normalized database design. A number of enhancements, especially in the last few releases, have greatly improved performance for data warehouse workloads which are typically de-normalized.

Normalization is the process of removing redundancies from the data. Transact-SQL queries then recombine the table data using relational join operations. By avoiding the need to update the same data in multiple places, normalization improves the efficiency of an application and reduces the opportunities to introduce errors due to inconsistent data. The SQL Server logical architecture defines how the data is logically grouped and presented to the users. The core components in this architecture are:

- **Tables**: Tables consist of one or more data pages. The table columns define the number and type of data that may be stored in the table. Each instance of the columns is stored as a row. The rows of a table may be ordered on the data pages and stored on disk according to the value of a clustered index key. Tables that do not have a clustered index key are stored on pages and disk in an unordered structure, also known as a heap. The storage of tables with a clustered index uses a binary tree or b-tree structure.
- **Indexes**: A non-clustered index defines a key value made up of one or more columns of a table and stored as a b-tree. Additional data from the table that is not part of the index key can also be included in the b-tree. An index can speed up access to data when the data can be searched for by the value of index key. Additional performance gains can be attained if all the data required by the query is contained in the leaf of an index as either part of the key or non-key included columns. This prevents the relational engine from performing an additional lookup from the parent table.

  Indexes require additional storage and maintenance that can be non-trivial and can adversely impact Insert, Update, and Delete performance. Indexes that are rarely or never used by the optimizer have costs with no corresponding benefit.
- **Views**: A view is a virtual table or a stored query. Views primarily assist application developers by reducing the number and complexity of queries to retrieve commonly accessed data that requires multiple tables to be joined, sorted and or filtered. A view may also be indexed and potentially used by the optimizer to speed up data access.
- **Stored procedures**: A stored procedure is a group of Transact-SQL statements compiled and stored in a single execution plan. Coding business logic into stored procedures creates a single point of control to ensure that business rules are correctly enforced. The use of stored procedures and proper parameterization is considered a best practice because of the execution plan efficiency, and prevention of SQL Injection

  Stored procedures can also improve performance through the reuse of cached execution plans. SQL Server has an efficient algorithm to find any existing execution plans for any specific SQL statement. Cached execution plans can become stale or not be optimized for all values of user supplied input parameters. There have been enhancements to how parameters are treated for cached plans such as optimize for unknown optimizer hint. SQL Server provides

numerous performance monitor counters and dynamic management views that you can access to determine if your stored procedures affect performance positively or negatively.

- **Constraints**: Constraints are commonly used to enforce data integrity in a database including referential, data, and unique keys or indexes.
- **User-defined functions**: Like functions used in other programming languages, SQL Server supports user-defined functions as named routines that accept parameters, perform an action, such as a complex calculation, and return the result of that action as a value. The return value can either be a single scalar value or a result set.
- **Triggers**: A trigger is a stored procedure that automatically executes when an event occurs in the database server. DML triggers execute when a user tries to modify data through a data manipulation language (DML) event. DDL triggers execute in response to a variety of data definition language (DDL) events. These events primarily correspond to Transact-SQL CREATE, ALTER, and DROP statements, and certain system stored procedures that perform DDL-like operations.

Not all database designs follow strict normalization rules. A database that is used primarily for decision support (as opposed to update-intensive transaction processing) may not process many redundant updates. The database structure may be more understandable and efficient for decision support queries if the design is not fully normalized. De-normalization can also help the query optimizer be more efficient for typical data warehouse queries.

The impact of database design on performance cannot be overstated. Databases that are not normalized are a more common design problem for OLTP workloads than having data structures that are over-normalized. Starting with a normalized design and then selectively de-normalizing tables for specific reasons may be the best strategy.

SQL Server physical components

The SQL Server physical components determine how the data is stored in the file system of the operating system. The selection of the number and types of table columns and index design has a major impact on the requirements for physical storage.

Database file types

SQL Server uses three types of files:

- **Primary data files**: Every database has one primary data file that stores data as well as information about other files used by the database.
- **Secondary data files**: A database can have zero or more secondary data files. Secondary data files are not required, and a database can have many secondary files or none. By convention, a secondary data file has an .NDF extension.
- **Log files**: Each database has at least one or more log files independent of the number of data files. Log files store the write ahead transaction log information that is needed to recover transactions for the database. By convention, a transaction log file has an .LDF extension.

Data files

Data files store the 8K pages used by SQL Server. A SQL Server page is the basic unit of logical data storage. A page begins with a 96-byte header that contains system information about the page. The disk space allocated to the primary or secondary data files (MDF or NDF) is logically divided into pages.

The most common types of pages that can be allocated to a data file are:

- Data pages
- LOB pages
- Index pages
- Page free space (PFS) pages
- Global allocation map and shared global allocation map (GAM and SGAM) pages
- Index allocation map (IAM) pages
- Bulk change map (BCM) pages
- Differential change map (DCM) pages

Figure 2 shows the relationship of these major page types in a data file:

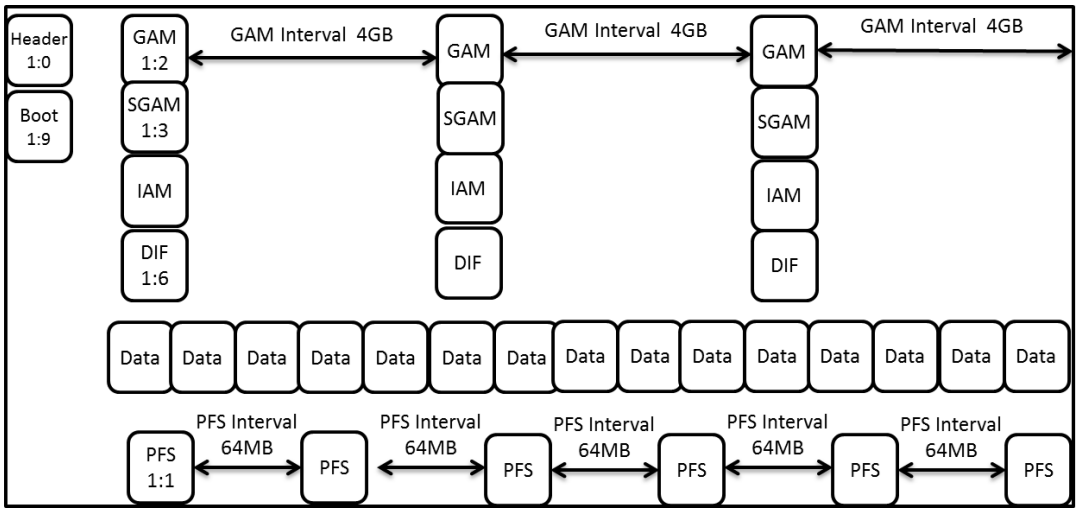

Figure 2. SQL Server data file internals

Extents

Extents are the basic units for allocation of space. Each extent has eight physically adjacent pages (64 KB). A new table or index is usually allocated pages from mixed extents. Uniform extents are used for subsequent allocations after the object grows beyond eight pages.

Some SQL Server features use extents. For example, database snapshots allocate new space in their associated NTFS sparse files using extents, even if only one page in the extent has changed. This allows SQL Server to put pages for subsequent changes from that extent in an adjacent location on disk to improve performance when reading from the sparse file. Also, when using differential backups, SQL Server

uses the differential change map pages in the data file to identify any extents that have been modified since the last full back. The backup engine then copies those changed extents to the differential backup file. On restore, the full backup is used to create a new copy of the database and then changed extents that are stored in the differential backup are used to overwrite the extents from the full backup that have changed.

File groups

Every database has a PRIMARY file group. Most databases have only the PRIMARY file group and one data file. User-defined file groups can be created to group data files together for administrative, data allocation, and placement purposes. A file group can contain one or more data files.

At any time, one file group is designated as the default file group. When objects are created in the database without being assigned to a file group, they are assigned to the default file group. The files in the default file group must be large enough to hold any new objects not allocated to other file groups. The PRIMARY file group is the default file group unless it is changed by using the ALTER DATABASE statement.

Historically, when SQL Server was deployed on servers that had a limited number of direct-attached disks, DBAs used multiple file groups and files to spread I/O across physical disks. It was common to create large tables and their corresponding indexes on different file groups so that the files could be allocated to different physical disks.

With the widespread adoption of RAID controllers and intelligent storage arrays, the added complexity of such detail object placement does not lead to better performance. There are, however, several reasons to use multiple files and file groups for a database including:

- Use of In-Memory tables requires the creation of a Memory Optimized file group.
- Filestream requires a filestream file group.
- Create multiple equal sized files in one file group because of a high page allocation rate. SQL Server will spread new allocations for objects created this way using a proportional fill algorithm. This technique is widely used for TEMPDB data files for instances that have high object creation and deletion rates. This can alleviate wait times associated with page latch waits on the data file allocation pages. It can also be necessary for user databases that have similar characteristics.
- Separate table and index partitions into read/write and read-only file groups for data warehouse applications. Read-only file groups only need to be backed up once. Subsequent backups can ignore the read-only file groups because that data does not change.
- SQL Server Enterprise Edition includes a feature that allows for piecemeal recovery of the database using multiple file groups. On restore, SQL Server Enterprise Edition can make the database available after the primary file group has been brought online. The other file groups can be brought online in any order.

- Use file group for partitioned table management. Partitions can be assigned to different file groups using different classes of storage. Partitions can be switched in and out of the table for better data loading and archiving.

Transaction logs

The transaction log records changes made to a database and stores enough information to allow SQL Server to recover the database. Recovery reconciles the data in the data files with changes recorded in the transaction log. The recovery process happens every time the server instance is restarted and optionally when a database or log restore occurs.

Physically, the transaction log consists of one or more files configured as a "circular log". If multiple log files exist you can think of them as being a single concatenated space. There are no parallel log operations when multiple files are used and typically no performance advantages. The use of multiple log files usually occurs when the existing log file needs to be extended but there is no allocated space on the Windows device. By adding another log file, the additional space is available to the log writer.

The buffer manager in the storage engine guarantees that the log file will be written to before the change is made to the database (called write ahead logging). Writes to the log are asynchronous, however, the storage engine must receive a successful response from the log write operation at the end of a transaction before acknowledgement is made to the client that the request was successful. The log write buffer can hold up to 60K of log data. The buffer is flushed at the completion of each transaction or when the buffer is full, whichever occurs first.

Information in the log is stored in variable length records that are uniquely identified by a Log Sequence Number. The log file contains information regarding:

- The start and end of each transaction
- Data modifications
- Extent and page allocations and de-allocations
- Creation and elimination of a table or index

Each log file consists of a number of virtual log files based on the initial size of all transaction log files and the growth increment set for auto expansion. Virtual log files are the smallest unit of truncation for the transaction log. When a virtual log file no longer contains log records for active transactions, it can be truncated and the space becomes available to log new transactions. Figure 3 shows how circular logging uses multiple virtual logs in a single physical file.

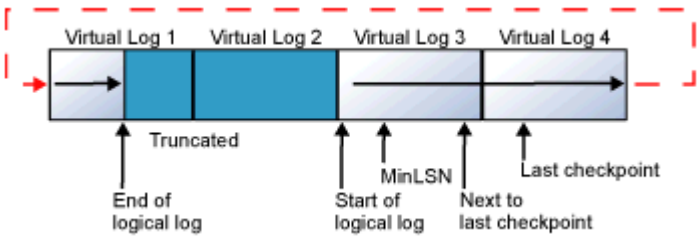

Figure 3. Circular logging

Windows disk and volume management

The SQL Server storage engine is dependent on the Windows operating system to make available disk and volumes for the placment of data and log files. Windows manages all I/O operations to those files.

The performance of a storage system used by SQL Server is dependant on various configuration parameters that are applied at the partition, disk, controller, SAN, RAID, and device driver levels. Review these configuration parameters with the storage system vendor to ensure that they are the appropriate settings for SQL Server I/O patterns.

The order of the I/O operations associated with SQL Server must be maintained. The system must maintain write ordering or it breaks the write ahead logging protocol discsussed above.

Any I/O subsystem that supports SQL Server must provide stable media capabilities for the database log and data files. If the system has a non-battery backed or mirrored cache it is not safe for SQL Server installations.

Windows disks

Microsoft recommends that you run SQL Server on computers with the NTFS file format. Installing SQL Server 2016 on a computer with FAT32 file system is supported but not recommended as it is less secure than the NTFS file system and limits the amount of data that can be written to the file. SQL Server Setup blocks installations on read-only, mapped, or compressed drives.

Windows volumes

Windows volume partition styles include master boot record (MBR) and globally unique identifiers (GUID) partition table (GPT):

- **MBR**: The legacy partitioning style, which allows a maximum of four partitions. The partition table is saved only at the beginning of the disk.
- **GPT**: A partitioning style capable of managing partitions larger than 2 TB. Its partition table is saved in multiple locations. It can be easily recovered if any partition is corrupted.

Two types of disk modes are supported:

- **Basic**: The most basic disk, which contains primary partitions and if needed, extended partitions. Basic mode features include:
  - **Primary partition**: A standard bootable partition.
  - **Extended partition**: A non-bootable partition. This is the fourth partition on a basic MBR disk, which holds logical partitions, thus allowing more than four partitions.
  - **Logical partition**: A non-bootable partition in each of the partitions contained within the extended partition.
  - **Extensible firmware Interface (EFI)**: Used to store boot files on EFI-compatible systems.

  - **Microsoft System Reserved (MSR)**: Only available on GPT basic disks, used to reserve space for future use.

- **Dynamic**: A native host-based logical volume manager, responsible for aggregating disks into logical volumes with multiple options. It creates two partitions; one contains all dynamic volumes and the other is hidden and contains the Logical Disk Manager (LDM) database. This database is replicated across all dynamic disks on the system so it is recoverable. It can host up to 2,000 dynamic volumes (a maximum of 32 is recommended). Dynamic mode features include:
  - **Simple**: Standalone volume
  - **Striped**: Like RAID 0, a striped volume writes a data block to both disks. The volumes integrating this arrangement must be the same size.
  - **Spanned**: Like RAID 0, with concatenated volumes. If the disk fails, only part of the data will be lost. The volumes are not required to be of the same size. It has lower performance than stripped volumes with the same number of disks.
  - **Mirror**: RAID 1
  - **RAID**: RAID 5

Note: Because EMC arrays such as the VNX family and VMAX systems provide storage RAID protection, avoid dynamic disks, as this option complicates the management of storage and local and remote disaster recovery (DR).

Storage types for data files

The supported storage types for SQL Server data files are:

- Local disk
- Shared storage
- SMB file share

Local disks can be direct attached (DAS), Fibre Channel, or iSCSI. They all are considered block storage. The file system is installed and managed by the local operating system. This is the most common form of storage for SQL Server.

iSCSI is a SCSI transport protocol for mapping block-oriented storage data over TCP/IP networks. Microsoft supports SQL Server when it is deployed on Internet small computer system interface (iSCSI) technology components that have received the "Designed for Windows" Logo Program qualification. SQL Server installations that use iSCSI will require these iSCSI hardware devices in addition to the network adapters that are required for typical network communications.

When you deploy SQL Server in an iSCSI environment, Microsoft recommends that you use caution. A network may introduce components that are not typically viewed as high speed I/O paths.

SQL Server failover clustered instances require shared storage. Disks presented as drive letters or mount points must first be claimed by the Windows Failover Clustering Role to support disk arbitration during a failover.

In network attached storage (NAS), the file system is managed by the remote host or service. Server Message Block (SMB) 3.0 is a network file sharing protocol that allows applications on a computer to read and write to files and to request services from server programs in a computer network. A Windows File Server or a third party SMB storage device can host SMB storage. SMB 3.0 was introduced in Windows Server 2012.

SQL Server 2012 and later versions support both virtualized disk (VHD/VHDX) and databases directly hosted on SMB 3.0 shares. The shares can be presented to Windows Server 2012 and later version or multiple cluster servers. EMC provides full support for SMB3.0 storage topology for SQL Server.

Chapter 3 Database Layout, I/O, and Bandwidth

This chapter presents the following topics:

Overview

Understanding SQL Server I/O patterns and characteristics is critical for designing and deploying SQL Server applications. A properly configured I/O subsystem can optimize SQL Server performance.

There are two types of generic SQL Server database workloads: OLTP and data warehouse/OLAP. A specific user database can generate an I/O workload that is very different from those in the standard benchmark. The only way to determine I/O performance needs is to analyze the database under a typical load in real time.

OLTP

OLTP workloads produce numerous concurrent transactions with significant random I/O reads and writes. OLTP databases change constantly. Most ad-hoc applications generate OLTP workload.

According to the TechNet topic *SQL Server Best Practice*, OLTP database workloads would typically have the following patterns:

- Both reads and writes issued against data files are generally random in nature.
- Read activity (in most cases) is constant in nature.
- Write activity to the data files occurs during checkpoint operations (the frequency is determined by recovery interval settings).
- Log writes are sequential in nature with a varying size depending on the nature of the workload (the sector is aligned up to 60 KB).
- Log reads are sequential in nature (the sector is aligned up to 120 KB).

OLTP databases may have many write activities that place pressure on the I/O subsystem, particularly on the log LUN, because the write goes to the transaction log first.

A typical OLTP system has a large number of concurrent connections actively adding and modifying data, for example, in an online airline reservation system. An OLTP system requires frequently backed up transaction logs and places further demands on the I/O subsystem.

In configurations that use the transactional replication, after the initial snapshot takes place, subsequent data changes and schema modifications made at the publisher are delivered to the subscriber, driving more read activity for the transaction log on the publisher database.

Index usage is another factor that affects the I/O subsystem. Heavily indexed OLTP systems used for operational reporting can support high concurrency with low latency to retrieve a small number of rows from datasets that contain very little historical data. The volatility of transactions within an OLTP system might require frequent index maintenance that places heavy read and write requests on the I/O subsystem.

OLTP systems usually generate a large number of input/output operations per second (IOPS).

Data warehousing/OLAP

Data warehousing is often the basis of a decision support system (DSS) or a Business Intelligence system. It is a repository of an organization's data, designed to facilitate complex analytical query activities using very large data sets for reporting and analysis. Data warehouse databases are OLAP-type databases, which typically use complex analysis with aggregated or summarized data in the data warehouse.

Data in the data warehouse system is usually static with sequential read and very little write activity, except for typical batch updates. I/O bandwidth is more important than IOPS. The typical workload in a data warehouse is I/O intensive with operations such as large data load and index build, and queries over large volumes of data. The underlying I/O subsystem for the data warehouse should meet these high bandwidth requirements.

The ideal I/O characteristics for a data warehouse are:

- Sequential reads and writes, generally the result of table or index scans and bulk insert operations
- Non-volatile data, larger historical datasets
- A light index in the fact table
- Low concurrency
- High tempdb activity
- Varied I/O size: typically larger than 8 KB. Read-ahead is any multiple of 8 KB up to 512 KB. Bulk load operations are any multiple of 8 KB, up to 128 KB.
- When using columnstore indexing, a database file I/O size that is much larger than 256 KB

A key design consideration for an efficient data warehouse storage solution is to balance the capabilities of the data warehouse system across the compute, network, and storage layers.

For example, the compute layer should be capable of processing data at bandwidth rates the storage can provide at comfortable utilization levels. In turn, the networking of compute and storage layers should be sufficient to sustain the maximum permissible throughput between compute and storage layers. Ideally, to ensure a cost-efficient data warehouse solution, one element of the solution should not have an excessive capability over the other.

When designing a data warehouse, estimate how much I/O bandwidth a given server and host bus adapter (HBA) cards can potentially use, and ensure that the selected I/O configuration can satisfy the server requirement.

A well-designed data warehouse system optimizes the storage system for scanning-centric operations; the server CPU can receive and process the data delivered by the storage at the same bandwidth. Because the queries in the data warehouse can fetch millions of records from the database for processing, the data is usually too large to fit in memory. A good storage design should quickly locate and deliver the data from disk for the processors to perform the aggregation and summarization.

I/O patterns

Table 2 summarizes the I/O patterns involved in each type of database.

Table 2. I/O patterns of different workloads for a SQL Server database

<table>
<tr><th>I/O types and characteristics</th><th>Database file OLTP</th><th>Decision support system (data warehouse, OLAP)</th></tr>
<tr><td>Data files</td><td>• Smaller random I/Os (8–64 KB)
• High proportion of reads compared to writes (typically 90/10 to 70/30 read/write ratio)</td><td>• Larger sequential I/Os (mostly 64 KB, can be more than 256 KB with columnstore index)
• Low proportion of writes compared to reads, sometimes read-only</td></tr>
<tr><td>Database log file</td><td colspan="2">• Small, highly sequential I/Os (multiples of 512 bytes up to 60k)
• Almost exclusively writes, with occasional reads during large rollbacks or log backups</td></tr>
<tr><td rowspan="2">tempdb data file</td><td colspan="2">• Varying size depending on usage (usually larger I/O, typically does not exceed 64 KB)
• Serial or random I/Os, a given workload might be somewhat sequential, many workloads running simultaneously may give tempdb a random I/O appearance
• Usually half writes and half reads</td></tr>
<tr><td>• Tempdb activity varies.
• Usually is not very active with low performance demand.
• Can be very active for frequent reporting and large table joins.</td><td>Tempdb can have high performance demands; consider using flash disks or server-side flash storage card.</td></tr>
</table>

Database partitioning

SQL Server table partitioning allows ranges of rows in the same table or index to be stored on disk in distinct file groups and files. Data is partitioned using a partition function based on the range of values in a chosen column. For example, a table can be partitioned based on a date field with the range of rows in a given year making up each file group. Table 3 shows an example of a partitioned sales table.

Table 3. Table partitioning example

Table name	File group name	Files	Disk type	Usage
Sales Table - Partition Function (Year)	FILEGROUP 2015	File 1	Flash	Active Data
		File 2	Flash	
	FILEGROUP 2014	File 3	Flash	
		File 4	Flash	
	FILEGROUP 2013	File 5	SAS	Historical Data
		File 6	SAS	
	FILEGROUP 2012	File 7	SAS	
		File 8	SAS	

Table partitioning offers the flexibility to physically store data based on the I/O requirements for each partition. Historical, less frequently accessed data can be stored on slower cheaper storage without compromising the performance requirement that may be required for more active partitions, which can be placed on faster storage.

Table partitioning is transparent at the database level because the logical table can be queried as if it were a single unit. However, the SQL Server relational engine uses table partitioning to optimize a query plan when feasible. Partitioning reduces the amount of data SQL Server has to scan to complete a query.

Table partitioning also helps database maintenance operations including index rebuilds, statistics updates, data compression, data loading, and backup and restore operations that can all be performed on subsets of data as required. There are some limitations, for more information refer to the MSDN topic Partitioned Tables and Indexes.

Reading pages

There are two types of I/O reads from the SQL Server database engine:

- **Logical read**: Occurs when the database engine requests a page from the buffer cache
- **Physical read**: Copies the page from the disk into the cache if the page is not currently in the buffer cache

The read requests are controlled by the relational engine and optimized by the storage engine. The read-ahead mechanism anticipates the data and index pages needed for a query execution plan and brings the pages into the buffer cache before being used by the query. This mechanism makes it possible to overlap computation with I/Os to make full use of both CPU and disk to optimize the performance.

Writing pages

There are two types of I/O writes from an instance of the database engine:

- **Logical write**: Occurs when data is modified on a page in buffer cache
- **Physical write**: Occurs when the page is written from buffer cache to the disk

Both page reads and page writes occur at the buffer cache. Each time a page is modified in the buffer cache, it is marked as "dirty." A page can have more than one logical write before it is physically written to disk. The log records must be written to disk before the associated dirty page is written to disk. To ensure data consistency, SQL Server uses write-ahead logging to prevent writing a dirty page before the associated log record is written to disk.

Figure 4 illustrates the page write operation in SQL Server.

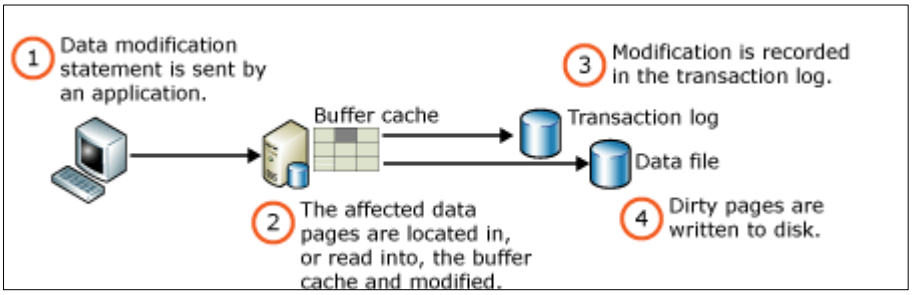

Figure 4. Page write operation in SQL Server

A dirty page is written to a disk in one of the following ways:

- **Lazy writing**: A system process that keeps free buffers available by removing infrequently used pages from the buffer cache. Lazy writing first writes dirty pages to the disk.
- **Eager writing**: This system process writes dirty pages with non-logged operations such as bulk insert or select.
- **Checkpoint**: The checkpoint operation periodically scans the buffer cache for database pages and writes all dirty pages to the disk.

The lazy writing, eager writing, and checkpoint processes use asynchronous I/O that allows the calling thread to continue processing while the I/O operation takes place in the background to maximize both CPU and I/O resources for the appropriate tasks.

Log manager

The log workload is the I/O against the transaction log; it usually has sequential writes and requires low latency for high scale transaction workloads. Transaction log file writes are synchronous for a given transaction, because SQL Server flushes all updates associated with a committed transaction to the log before the user thread can start the next transaction.

The physical transaction log file is internally divided into multiple virtual log files. Those virtual log files might affect system performance if the size and growth incremental values that define log files are too small. If the physical log file grows to a large size because of many small increments, they will have many virtual log files. As a result, this can slow down database startup and also log backup and restore operations.

EMC recommends that when you create a database, assign the size value of log files approximately to the final size as required to avoid too many small increments, and also set a relatively large growth increment value. Refer to *Transaction Log Architecture* for more details.

Indirect checkpoints

Indirect checkpoints, introduced in SQL Server 2012, allow more granular and accurate control of a database checkpoint operation. Unlike automatic checkpoints, the recovery interval can be set at a database level rather than at an instance level. The recovery interval for indirect checkpoints is more accurate because it factors in the I/O cost of redo operations. Long running transactions that require a large number of undo operations can still increase the recovery time.

Indirect checkpoints can reduce checkpoint I/O spiking by writing pages to disk more frequently. This increase in database writes may affect OLTP performance.

For example, Figure 5 shows the performance and I/O impact of both automatic (default setting) and indirect checkpoints when running certain SQL Server OLTP workloads.

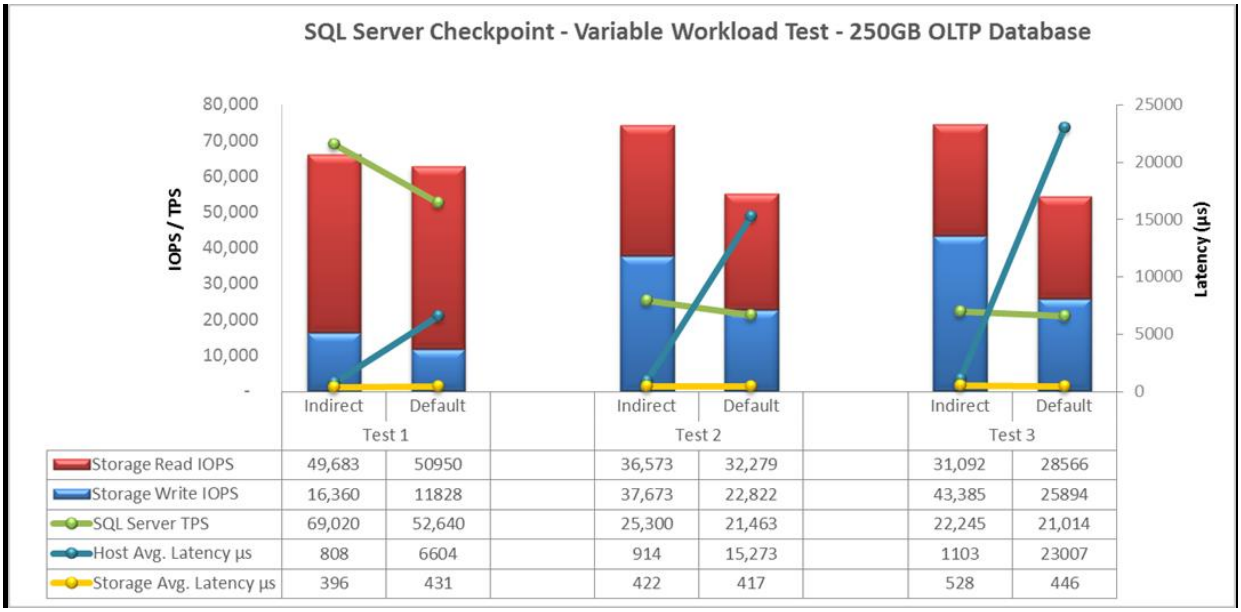

Figure 5. Indirect checkpoints compared to automatic checkpoints

Note: The reference result in Figure 5 is tested under workloads derived from an industry standard modern OLTP benchmark. The recovery interval for indirect checkpointing is set to **60** seconds. The result may vary based on recovery interval settings and the actual type of SQL Server workloads.

In Figure 5, while indirect checkpoint resulted in additional I/O operations, it offered superior performance in terms of database transactions per second. Compared to the default checkpoint settings, indirect checkpoint can also eliminate I/O spiking and avoid potential performance bottleneck.

Index operations and fill factor settings

SQL Server automatically writes data modifications (insert, update, and delete) to disk. A page split occurs when modifications to a data row result in that row not fitting on a data page. The page is split into two separate pages. This can result in both logical and physical fragmentation:

- **Logical fragmentation** means data pages are physically stored out of order to the key value in the table or index.
- **Physical fragmentation** means free space exists internally within a data page. The fill factor setting determines the level of free space left on a data page when it is created or rebuilt.

A high level of fragmentation in a database can increase the amount of I/O operations and affect the I/O pattern when accessing data. Figure 6 shows the I/O pattern when an identical SQL Server query runs against a fragmented index and then the same query runs again after the index is rebuilt. Depending on the storage array and disk, fragmentation affects workload performance and capacity.

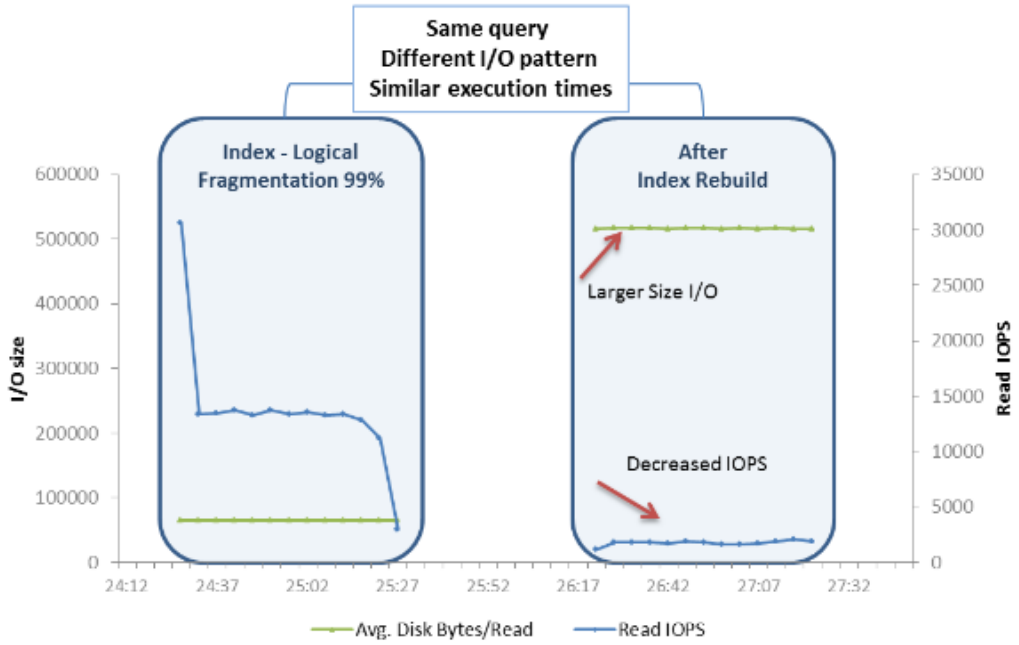

Figure 6. Impact of index fragmentation on performance

Index maintenance is the process of reordering and compacting data so it is stored and can be accessed efficiently. Depending on the level of fragmentation, either an index rebuild or an index re-organization can be executed. The level of fragmentation can be calculated using SQL Server Dynamic Management Views (DMVs):

- Index reorganization is an online operation and requires no additional space in the data file to complete.
- Index rebuilds can be online or offline. The database data files need enough free space to accommodate the size of the largest index in the database. This is because SQL Server builds a new index structure and keeps the old one until the operation is complete. Tempdb can be used to store the intermediary sort results.

Chapter 4 Storage Sizing and Provisioning Best Practices

This chapter presents the following topics:

Overview

Storage design is one of the most important elements of a successful SQL Server deployment. To achieve a storage design for optimal reliability, performance, and ease of use, follow the recommended storage guidelines.

This section provides general best practices for deploying SQL Server on EMC storage, such as XtremIO All-Flash Array, ScaleIO software-based storage, Symmetrix VMAX series storage, VNX unified storage, and recommendations for specific EMC storage array features that could affect SQL Server.

Because the virtualization of a SQL Server environment requires its own set of considerations, this section also includes guidance on this subject.

SQL Server general best practices for server and storage

General SQL Server storage best practices

For the SQL Server configuration, begin by considering the following basic requirements:

- **OS/SQL Server binaries**

  In a typical SQL Server implementation, the server is dedicated for SQL Server and binaries are on the same LUN as the operating system. Follow Microsoft's recommendation for the operating system type and SQL Server version and consider the overhead for applications that you need to install on that server.

  Typical LUNs for the operating system /SQL Server binaries/system databases is 60 GB to 120 GB.

- **System databases**

  In most environments, system databases are not frequently changed or modified and they can be on the same LUN as the operating system.

- **Logs for user databases**

  Logs for user databases typically need low IOPS (mostly sequential writes). Even with replication such as AlwaysOn Availability Group (AAG), the IOPS needed on these LUNs are typically not very demanding. Thus, the log LUNs are usually configured with Fibre Channel (FC) disks (in a storage pool, it can be pinned to the FC tier) that can satisfy the capacity need with at least 10 percent extra space.

- **tempdb**

  In an OLTP environment, tempdb may not be very I/O demanding and can follow the same design principle for logs. In this case, they can usually be in the same pool with SQL Server database log LUNs.

  When there is scheduled or ad-hoc reporting or large table joins, tempdb can experience heavy usage. You must measure the SQL Server system needs to determine the tempdb usage.

A tempdb in a data warehouse or for OLAP workloads is typically under very intensive I/O demands and warrants special attention in these environments. The tempdb design in these environments should follow the database design principle for sizing and placement if needed.

- **User database**

  User database LUNs are typically the main focus for storage design. LUN types vary depending on performance and capacity requirements, as well as the workload type.

Some key SQL Server storage best practices are listed in the following sections.

Basic best practices for SQL Server

General SQL Server best practices:

- Select the **Lock pages in memory** policy for the SQL Server start account to prevent SQL Server from swapping memory.
- Pre-allocate data files to avoid **Autogrow** during peak time.
- Set **Autoshrink** to **Off** for data and log files.
- Make data files of equal size in the same database—SQL Server uses a proportional fill algorithm that favors allocations in files with more free space.
- Perform routine maintenance with index rebuild or reorganization with the command: **dbcc checkdb**.

Note: For XtremIO, there is no need for an index rebuild or reorganization. For more information, refer to XtremIO best practice section.

File group and file considerations

File group considerations in SQL Server:

- File groups can be accessed in parallel; placing file groups on different set of disks or storage pools can improve performance.
- Organize SQL Server data files with similar performance and protection needs into a file group when designing a database.
- Create 0.25 to 1 data files per file group for each CPU core for tempdb that under excessive usage or intensive workloads.
- Create one log file in a typical environment. More log files will not improve performance.

Refer to the Microsoft TechNet topic *Using Files and Filegroups* for more information.

Basic best practices for storage

The following are some high-level basic best practices for storage design. The details are discussed in General hybrid storage considerations.

- Plan for performance, capacity, and protection. Table 4 lists the response times for data file and log files.

Table 4. Response times for data file and log files

I/O response time	Data file	log
Excellent	Less than 5 ms	1 to 5 ms
Very good	Less than 10 ms	Less than 5 ms
Acceptable	10 to 20 ms	5 to 15 ms
Needs investigation and improvement	Greater than 20 ms	Greater than 15 ms

- When creating a volume in Windows, set the Windows allocation unit to 64 K for SQL Server database and log LUNs.
- For optimal performance with a predictable level of service, place tempdb, data, and log files on separate LUNs.
- To leave room for data growth, avoid exceeding 80 percent capacity of the LUNs for database files.
- Use up-to-date manufacturer-recommended HBA drivers.
- Ensure that storage array firmware is up-to-date.
- EMC strongly recommends using multipathing for availability/redundancy and throughput optimization, especially in iSCSI/file-based configurations.

General hybrid storage considerations

In this section, general best practices and recommendations are included for the hybrid storage platforms, including VNX unified storage and VMAX storage systems.

Performance versus capacity considerations

When deploying SQL Server, always consider performance, protection, and capacity requirements.

For typical OLTP workloads, throughput measurements in IOPS typically outweigh the capacity requirement for database files and log files. Tempdb files are often capacity bound because of the low I/O nature of the workload.

In an OLAP environment, bandwidth measurements in megabytes or gigabytes are more dominant for database files, while tempdb files are likely to require higher throughput (IOPS).

When designing storage for different type of workloads, consider the workload type and its typical I/O pattern for database, log, and tempdb files. Calculate both performance and capacity requirements to ensure that both are satisfied.

Disk type selection

One of the first key decisions you must make when designing SQL Server storage is selecting the media type that best match your requirements. The types of disks that are appropriate for your SQL Server deployment depend on a number of factors, including your database size and IOPS requirements.

Table 5 shows the disk types that EMC offers with its VNX family of unified storage and Symmetrix VMAX series storage.

Table 5. Disk types offered by EMC

Disk type	Characteristics	Selection consideration
Flash	Flash drives have the highest I/O speed with low power consumption.	In general, flash drives can be used as follows: • In the storage array as part of the automated storage tiering features, such as EMC FAST VP or FAST Cache to handle any unanticipated I/O spikes EMC also provides all flash arrays such as XtremIO™ and VMAX3 or VNX configured as an all-flash for the most demanding SQL Server environment.
Fibre Channel (FC)	Reliable disk drives with high read/write speeds.	Ideal for high I/O requirements but might not be appropriate for high capacity requirements.
Serial Attached SCSI (SAS)	An improvement over traditional SCSI drives, SAS disks provide high capacity with moderate I/O speed.	Highly suitable for SQL Server environments with high IOPS requirements.

SATA	Large capacity disks with less demanding I/O speed.	Appropriate for large databases with low I/O requirements. Most suitable for data warehouse and SharePoint content database.
Near-line SAS (NL-SAS)	As with SATA disks, NL-SAS disks are a good fit for less demanding I/O but large capacity requirements.	NL-SAS disks can support large databases at a relatively low cost. NL-SAS disks are typically the best choice for larger databases with low I/O profiles.

Follow these general rules when selecting disk types:

- For low IOPS, acceptable disk latency, and high database capacity requirements, use SATA or NL-SAS disks.
- For high IOPS, low disk latency, and high database capacity requirements, use large capacity FC or SAS disks.
- For higher IOPS and very low disk latency requirements but smaller database capacity requirements, use flash drives in storage tiering or FAST Cache.

Different disk types support different IOPS with the same latency requirement. Consider this when calculating disk requirements for your environment. Table 6 provides random disk IOPS data from the most recent SQL Server validation on EMC VNX and VMAX storage. These results are subject to change, based on future testing.

Note: EMC recommends using values from Table 6 when calculating IOPS requirements for SQL Server deployment on VNX and VMAX storage arrays. These numbers give a baseline of typical acceptable performance, as shown in Table 4. For applications that require better performance, add more disks or use array caching such as FAST Cache.

Table 6. IOPS for 8 KB random read I/O on various disk types on EMC storage arrays

Disk type	IOPS per disk
15 K rpm SAS	180
10 K rpm SAS	140
7.2 K rpm NL/SAS	70
Solid-state disk	3,500

Pools and RAID types considerations

Storage pools are virtual constructs that enable data to move dynamically across different tiers of drives (from high performance to lower cost and higher capacity, and conversely) according to the data's business activity. With EMC storage systems, storage pools are fully automated and self-managing.

Pool-based provisioning provides benefits similar to metaLUNs striping across many drives but, unlike metaLUNs, storage pools require minimal planning and management efforts.

Storage pools support the same RAID protection levels as RAID groups: RAID 5, RAID 6, and RAID 1/0. Multi-tiered pools with different RAID and disk types can be in the

same storage pool. Storage pools also provide advanced data services such as FAST VP, compression and deduplication, and data protection options such as snapshot functions.

Most SQL Server database environments can benefit from storage pool-based configurations.

The selection of an appropriate RAID type for your environment is another important decision for a successful implementation of SQL Server, especially for EMC VNX and VMAX platforms. Any RAID type can be used if enough disks are available to handle the I/O and storage capacity requirements. In general, RAID type decisions are based on a set of a given requirements. To select an appropriate RAID type for your environment, consider your specific requirements for performance, capacity, and availability.

EMC hybrid storage systems such as VNX and VMAX support RAID 1/0, RAID 5, and RAID 6 using flash, FC, SAS, NL-SAS, and SATA drives. Each RAID type provides different performance, capacity, and protection levels.

- RAID 1/0 provides data protection by mirroring data onto another disk. This produces better performance and has minimal or no performance impact in the event of disk failure. In general, RAID 1/0 is the best choice for SQL Server, especially if SATA and NL-SAS drives are used.
- RAID 5 data is striped across disks in large stripe sizes. The parity information is stored across all disks so that data can be reconstructed. This can protect against a single-disk failure. With its high write penalty, RAID 5 is most appropriate in environments with mostly read I/Os and where large databases are deployed. In the case of SSD flash drives, this performance concern is eliminated and most environments with flash drives can be configured as RAID 5 to support high I/O requirements with very low disk latency.
- RAID 6 data is also striped across disks in large stripe sizes. However, two sets of parity information are stored across all disks so that data can be reconstructed if required. RAID 6 can accommodate the simultaneous failure of two disks without data loss.

Table 7 shows RAID overhead, performance, and storage utilization information for each RAID type.

Note: The RAID overhead value becomes important when performing I/O calculations for the number of disks required. RAID 5 and RAID 6 have an impact on performance when a drive fails and must be rebuilt. In Table 7, the performance is compared using the same number and the same type of disks in the RAID configurations. Storage utilization is compared using the same disk types in the RAID configurations to generate the same IOPS with similar latency.

Table 7. RAID level performance characteristics

RAID level	Random Read	Random Write	Sequential Read	Sequential Write	RAID write overhead value	Storage utilization
RAID 1/0	Excellent	Excellent	Excellent	Excellent	2	Low
RAID 5	Excellent	Moderate	Good	Moderate	4	High
RAID 6	Good	Poor	Good	Moderate	6	Medium

Virtual provisioning storage considerations

EMC storage platforms offer virtual provisioning, generally known in the industry as thin provisioning. Thin or virtual provisioning can simplify storage management and reduce storage costs by increasing capacity utilization for many SQL Server use cases.

Virtual provisioning enables SQL Server to acquire more capacity than is physically allocated. The physical storage is allocated to the database "on demand" from a shared pool as it is needed.

Physical storage capacity is fully allocated during LUN creation for thick LUNs. While thin LUNs initially have less physical storage capacity allocated to them, the storage pool provides actual physical storage supporting thin LUN allocations when needed. Physical storage is automatically allocated only when new data blocks are written to the thin LUN.

Both thick and thin LUNs can provide required performance characteristics for any SQL Server workload.

Storage sizing best practices

SQL Server performance characteristics can vary substantially from one environment to another, depending on the application. These characteristics fall into two general categories: OLTP generates mostly random read workloads, and data warehouse generates mostly sequential read workloads. In an OLTP environment, use read/write IOPS (IO/s) for storage sizing. For a data warehouse environment, use bandwidth (MB/s) for storage sizing.

For accurate performance estimates, run tests in conditions as close to "real world" as possible. During these tests, use performance monitor logs to capture the characteristics (such as IOPS: reads/second and writes/second, bandwidth: MB/s) of the volumes used to store database files.

Before considering storage sizing for SQL Server, EMC strongly recommends that you use the following tools for the specific storage platform:

- For **XtremIO** all flash array sizing, refer to XtremIO Sizing Tool
- For **VMAX** storage system sizing, refer to VMAX Sizer Tool
- For **VNX** unified storage array sizing, refer to EMC VSPEX Sizing Tool

Notes:

- Do not use IOPS counters averaged over time as the basis for storage design. EMC recommends identifying the 90th percentile of the IOPS samples and designing for that performance level. This allows the system to respond well to spikes on demand.
- The IOPS requirements for a RAID group must be calculated independently for reads and writes.

Considerations for OLTP database sizing

Sizing for performance

Calculate the number of disks for performance requirements with the following formula:

$$Number\ of\ disks = \frac{Read\ IOPS + (Write\ IOPS \times\ RAID\ performance\ overhead)}{\text{IOPS per disk}}$$

Note: You might need to adjust the disk count to conform to the requirements for the selected RAID level. For example, a seven-disk RAID 1/0 set is not possible. In such cases an eight-disk RAID 1/0 set is required.

IOPS per Disk is the number of IOPS of the selected drive type.

Table 6 in Disk type selection provides the IOPS per disk for different disk types recommended for calculating the disk needs for RAID configuration and tiered storage configuration.

In a tiered storage configuration, you must also calculate the capacity requirements for different tiers. The following tools can help in tiered storage design:

- Tier Advisor for sizing

  You will need historical performance data from storage arrays.

- The EMC Workload Performance Assessment tool

  Also known as Mitrend, it shows the FAST VP heat map. For more information, refer to https://emc.mitrend.com.

Sizing for capacity

After the performance sizing is complete, you must consider the storage requirement for capacity. Although OLTP database sizing is typically bound with performance, verify the capacity requirement before the final design is complete.

Calculate the sizing for capacity based on the configuration in Table 8.

Table 8. Sizing for capacity

RAID type	RAID overhead for capacity	RAID overhead for performance (write penalty)	Minimal drives	Minimal disks required for the LUN size
1/0	1/2	2	4	2 × LUN size
5	4/5	4	3	5/4 × LUN size
6	4/6	6	4	6/4 × LUN size

Use the following formula to calculate the number of disks for capacity:

$$Number\ of\ disks = \frac{Total\ capacity\ need}{RAID\ overhead\ for\ capacity}$$

Note: When calculating the LUN size, consider the future growth of the database size. We recommend that you reserve at least 10 percent of the capacity for database file LUNs.

Final design

Use the larger number as the final sizing to satisfy both performance and capacity requirements.

Consideration for OLAP database sizing

For predictable bandwidth performance, EMC recommends a building block approach with consideration for the following:

- Targeted bandwidth
- Memory consumption
- CPU number
- Database sizing
- tempdb sizing

The design principles are as follows:

- **Design for bandwidth, and then for database capacity.**

  For OLAP databases, to achieve the goal of completing all the queries in order to generate timely reports, the storage design needs to meet the required bandwidth.

- **Check disk performance.**

  For workloads with sequential large read-only I/O (typically 64K or more), calculate the bandwidth for a given I/O size with the IOPS:

  $$Number\ of\ disks\ for\ tier = Average\ I/O\ size\ x\ read\ only\ IOPS$$

- **Calculate the storage requirement for database files.**

  Calculate the number of disks needed for performance as follows:

  $$Number\ of\ disks\ for\ performance = \frac{Required\ bandwidth}{Bandwidth\ per\ disk}$$

Calculate the number of disks needed for capacity as follows:

$$Number\ of\ disks\ for\ capacity = \frac{Required\ capacity}{RAID\ overhead\ for\ capacity}$$

Round up to the next logical drive count for its specific RAID type for each of the above calculations.

For tiered storage, calculate tier storage needs for the OLTP workload based on skew, as discussed in Best practices for FAST VP sizing.

- **Use the recommended 1:5 ratio for tempdb size**

  EMC recommends a 100 GB tempdb for every 500 GB database file in an OLAP environment. The actual tempdb size depends on the specific environment.

For details about the OLTP building block design, refer to the *SQL SERVER 2012 DATA WAREHOUSE: EMC VNX5500 HB (R32), VMware vSphere 5, Microsoft Windows 2012 White Paper*.

Hypervisor considerations

With current server technology rapidly reducing the cost of processing power, most physical server environments become under-utilized, yet some functions must be on separate servers for SQL Server applications to work properly. Virtualization can optimize data center resources by consolidating server resources to a few physical servers. This reduces power consumption, saves space, improves return on investment (ROI), increases manageability and flexibility, and introduces new high-availability options.

The Windows Server Virtualization Validation Program (SVVP), which is available on the Microsoft website, provides information about supported EMC storage for virtualizing SQL Server environments.

The virtualization of a SQL Server environment requires unique storage design best practices.

By virtualizing a SQL Server environment hosted on EMC storage platforms, customers can use features such as VMware vMotion and Microsoft Hyper-V live migration tools. These features enable virtual servers to move between different server hardware platforms without application disruption.

General virtualization guidelines

The following general guidelines apply to the virtualization of SQL Server.

General guidance

The general SQL Server virtualization design principles are:

- Design for performance, reliability, and capacity.
- Design for workload profiles when possible (such as OLTP and OLAP).
- Physical sizing still applies to calculate the disk number or FAST VP optimization.

- We recommend that you install EMC PowerPath® on physical Hyper-V or ESX hosts for load balancing, path management, and I/O path failure detection.
- HA/DR considerations: Disable migration technologies that save state and migrate. Always perform a live migration or completely shut down virtual machines.
- Disable hypervisor-based auto tuning features.

For optimal performance with Tier 1 mission-critical SQL Server instances:

- Follow the same guidelines as for a physical environment; separate LUNs for data and logs.
- Ensure that each ESXi or Hyper-V server has at least four paths (two HBAs) to the storage, with a total of four ports. Ensure that connectivity to the storage array is provided to both storage processors for VNX, or across multiple front-end directors on VMAX, or to both storage controllers for each XtremIO brick.
- Place SQL Server storage on separate disks from the guest OS (VHD/VHDX or VMDK) physical storage.

For acceptable performance with Tier 2 and lower service level instances:

- Virtual Hard Disks (VHDs) for virtual machine operating system LUNs can share a LUN at the hypervisor level (datastore for vSphere or the host level LUNs for VHD/ VHDX in Hyper-V) with the SQL Server database and log LUNs if ease of deployment is the main concern and performance is secondary.
- Multiple virtual machines or databases can share LUNs at the hypervisor level (datastore for vSphere or the host level LUNs for VHD/ VHDX in Hyper-V) if the HA/DR level is acceptable for the specific environment.

Refer to VMware vSphere environment and Microsoft Hyper-V environment recommendations for more details.

Hypervisor limits and considerations

The virtual machine best practices for VMware vSphere are:

- Size physical servers to accommodate the number of guests.
- For performance-critical instances, keep the number of physical cores and vCPUs in a 1:1 ratio.
  - Ensure that there are no overcommitted CPUs.
  - Do not exceed the size of the NUMA node on the physical server when sizing virtual machines. For details, refer to the VMware topic Using NUMA Systems with ESX/ESXi.
- Fully reserve RAM for SQL Server virtual machines to avoid any memory ballooning.
- Understand hypervisor limits (for Hyper-V refer to the TechNet topic *Hyper-V scalability in Windows Server 2012 and Windows Server 2012 R2* , for VMware refer to article *Configuration Maximums*). Configuration limits are shown in Table 9.

Table 9. Virtual machine configuration limits for vSphere and Hyper-V

	VMware vSphere	Hyper-V
Maximum memory	1 TB (ESXi 5.1, ESXi 5.5) 4TB (ESXi 6.0)	1TB (Windows 2012 R2, Windows 2012 R2 Hyper-V)
SCSI LUNs per virtual machine	60 (ESXi 5.1, ESXi 5.5, ESXi 6.0)	256 (Windows 2012 R2, Windows 2012 R2 Hyper-V)
Processor limits	32 vCPUs (ESXi 5.1) 64 vCPUs (ESXi 5.5) 128 vCPUs (ESXi 6.0)	64 vCPUs (Windows 2012 R2, Windows 2012 R2 Hyper-V)

- Use large pages in the guest (start SQL Server with a Trace flag—T834) for a dedicated 64-bit SQL Server to improve performance. For more information, refer to the Microsoft support topic *Tuning options for SQL Server when running in high performance workloads*.
- Enable the lock pages in memory privileges for the SQL Server Service account; for details refer to the TechNet topic *Enable the Lock Pages in Memory Option (Windows)*.

AlwaysOn technology considerations

For AlwaysOn Failover Cluster Instance (AlwaysOn FCI):

- Protection is necessary for tempdb in SQL Server. The tempdb file is re-created whenever a SQL Server instance starts. For SQL Server FCI configuration, always consider using unshared disk partitions for the tempdb whenever possible to reduce cost and bandwidth.

For AlwaysOn Availability Group (AlwaysOn AG):

- Spread AlwaysOn Availability Groups replicas across multiple physical hosts to minimize potential downtime in the event of physical server issues.
- If the SQL Server virtual machines in an AlwaysOn Availability Group are part of a hypervisor host-based failover clustering and migration technology, configure the virtual machines to not save or restore their state on the disk when moved or taken offline.

There are also limitations with both AlwaysOn FCI and AlwaysOn AG clusters for VMware, as listed in Table 10.

Table 10. VMware limitations for SQL Server clusters

Feature		Shared disk			Non-shared disk	
Microsoft clustering on VMware		MSCS with shared disk	SQL Server clustering	SQL Server AlwaysOn Failover Cluster instance (FCI)	Network load balance	SQL Server AlwaysOn Availability Group (AlwaysOn AG)
vSphere support		Yes	Yes	Yes	Yes	Yes
VMware HA support		Yes[1]	Yes[1]	Yes[1]	Yes[1]	Yes[1]
vMotion DRS support		No	No	No	Yes	Yes
Storage vMotion support		No	No	No	Yes	Yes
MSCS node limits		2 (5.1) 5 (5.5, 6.0)	2 (5.1) 5 (5.5, 6.0)	2 ((5.1) 5 (5.5, 6.0)	Same as OS/app	Same as OS/app
Storage protocols support	FC	Yes	Yes	Yes	Yes	Yes
	In-guest OS iSCSI	Yes	Yes	Yes	Yes	Yes
	Native iSCSI	Yes[2]	Yes[2]	Yes[2]	Yes	Yes
	In-guest OS SMB	Yes[3]	Yes[3]	Yes[3]	NA	NA
Shared disk	FCoE	Yes[4]	Yes[4]	Yes[4]	Yes	Yes
	RDM	Yes	Yes	Yes	NA	NA
	VMFS	Yes[5]	Yes[5]	Yes[5]	NA	NA

For more details about VMware limitations for SQL Server clusters, refer to VMware Knowledge Base.

[1] When DRS affinity/anti-affinity rules are used.

[2] vSphere 5.5 only.

[3] Windows Server 2012 Failover Clustering only.

[4] In vSphere 5.5, native FCoE is supported. In vSphere 5.1 Update 1 and 5.0 Update 3, two-node cluster configuration with Cisco CNA cards (VIC 1240/1280) and driver version 1.5.0.8 is supported on Windows 2008 R2 SP1 64-bit Guest OS. For more information, refer to the VMware Compatibility Guide for Cisco UCS VIC1240 and Cisco UCS VIC1280

[5] Supported in Cluster in a Box (CIB) configurations only. For more information, refer to the Considerations for Shared Storage Clustering section in this article.

VMware vSphere environment

General best practices for deploying SQL Server in a VMware vSphere virtual environment are:

- Deploy application virtual machines on shared storage. This allows the use of vSphere features like vMotion, HA, and Distributed Resource Scheduler (DRS).
- Create VMFS file systems from vCenter to ensure partition alignment.
- Add about five percent CPU requirements for hypervisor overhead.
- When using VMFS for data storage, format VMDK files as **eagerzeroedthick** (specifically for database and log files.)
- Consider installing VMware tools on each SQL Server virtual machine. Use **VMXNET3** paravirtualized adapter drivers to increase performance.
- Use multiple PVSCSI adapters and evenly distribute target devices.

  VMware Paravirtual SCSI (PVSCSI) adapters are high-performance storage drivers that can improve throughput and reduce CPU use. PVSCSI adapters are best suited for SAN environments, where hardware or applications drive high I/O throughput.

  VMware vSphere 6.0 introduces support for the PVSCSI adapter with virtual machines running Windows Server Failover Clustering (WSFC). This provides superior performance to the standard SCSI adapter. This enhancement helps deliver the storage I/O needed for the demanding SQL Server application workloads.

  Figure 7 shows that the SCSI controller type was changed to **Paravirtual** to improve driver efficiency. The default SCSI controller driver is LSI Logic SAS. LUNs are spread across all available SCSI drivers.

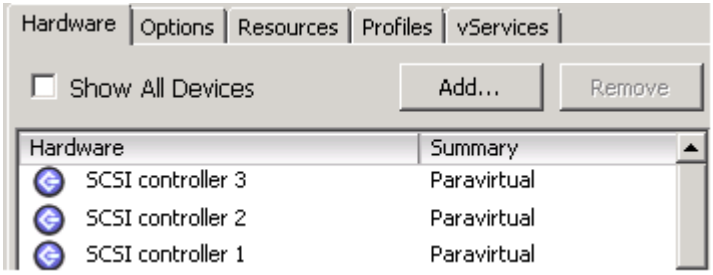

Figure 7. SCSI controllers

- Increase the queue depth of the HBA in the ESXi server to improve the throughput of HBA cards on ESXi servers. We observed a benefit of setting the queue depth of the HBA to 64 (which is the default value in ESXi 5.0, ESXi 5.1, ESXi 5.5 and ESXi 6.0). Results may vary in specific environments.

Notes:

- The highest queue depth configurable for the HBA ports in ESXi 5.0 is 128 and ESXi 5.1 is 256, if associated LUNs have dedicated ESXi server ports.
- For ESXi 5.0 servers with multiple virtual machines, the value of **Disk.SchedNumReqOutstanding** in VMware advanced options needs to match the queue depth.

For more information about ESXi server configuration, refer to the VMware knowledge base website and VMware ESX Scalable Storage Performance.

VMFS versus RDM

VMFS and RDM generally share similar performance characteristics, with RDM providing a slight edge. VMFS can be used in most environments unless there is a specific requirement for clustering and snapshot replication. Refer to the VMware website for details.

Table 11 compares VMFS and RDM in the VMware environment for SQL Server.

Table 11. VMFS versus RDM

VMFS	RDM
Better storage consolidation—multiple virtual disks and virtual machines per VMFS LUN; but still supports one virtual machine per LUN	Enforces 1:1 mapping between virtual machine and LUN
Consolidating virtual machines in LUNs—less likely to reach the ESX LUN limit of 255	More likely to hit the ESX LUN limit of 255
Managing performance: combined IOPS of all virtual machines in the LUN is less than the IOPS rating of the LUN	Not impacted by IOPS of other virtual machines
Limited LUN size (2 TB for vSphere 5.1 and 62 TB for vSphere 5.5)	Unlimited LUN size
Works well for most environments other than those requiring raw device mapping (RDM)	In ESXi 5.5, shared disk quorum or data must be provisioned to guest in PassThrough RDM mode only, when path selection policy is round robin (PSP_RR). Required for SAN management tasks such as backup and snapshots

vSphere queue depth considerations

In the VMware vSphere environment, consider optimizing the storage I/O queue depth for your EMC storage array to support heavy SQL Server workloads. Figure 8 shows the basic types of queues in vSphere:

- **World Queue:** per virtual machine queue
- **Adapter Queue:** per HBA queue
- **Device Queue:** per LUN queue

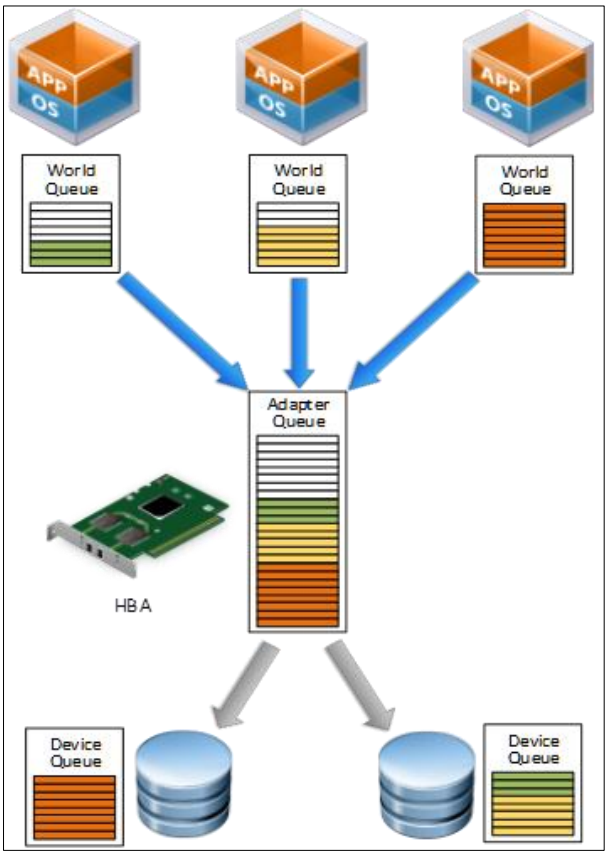

Figure 8. Storage I/O queue flow in VMware vSphere

The I/O request flows into the World Queue, which then flows into the Adapter Queue and, finally, into the Device Queue for the LUN to which the I/O is going. For more details, refer to the VMware blog post *Troubleshooting Storage Performance in vSphere – Storage Queues*.

Table 12 shows the support of VMware vSphere queue depth settings for EMC storage arrays. For SQL Server workloads with high performance demands, you may increase the default queue depth value of each setting to achieve performance benefits. For more details refer to Appendix A.

Table 12. VMware vSphere queue depth settings

Array	FC Adapter Policy I/O Throttle Count (per vHBA)	fnic_max _qdepth (global)	Disk. SchedNumReq Outstanding (per device/LUN)	Disk. SchedQuantum (global)	Disk. DiskMaxIOSize (global)	VAAI (global)
XtremIO	1024	128	256	64	4096	Enabled
VMAX	16 (default), Up to 256	32 (default)	32 (default)	8 (default)	32767 (default)	Enabled
VNX	16 (default) , Up to 256	32 (default)	32 (default)	8 (default)	32767 (default)	Enabled

Note: For XtremIO, refer to *EMC XtremIO Storage Array Host Configuration Guide* for detailed steps on setting queue depth with different HBA vendor implementations.

Note: For VMAX and VMAX3 recommendations with VMware refer to *Using EMC VMAX Storage in VMware vSphere Environments* for a complete and detailed list.

Microsoft Hyper-V environment

Storage options for SQL Server in Hyper-V

The storage options for SQL Server are:

- **VHDX** is the new virtual hard disk format introduced in Windows Server 2012 Hyper-V; it provides increased storage capacity and data protection.
- **VHD**, the Windows Server 2008 Hyper-V virtual hard disk, can be converted to VHDX in Windows Server 2012 Hyper-V.
- SQL Server database files and log files can reside on VHD or VHDX. VHDX can be created on a Cluster Shared Volume (CSV) to take advantage of the HA/DR features that Hyper-V provides.
- **CSV** is a feature of failover clustering, which was first introduced in Windows Server 2008 R2 for use with Hyper-V. CSV is a shared disk containing an NT file system (NTFS) volume that is made accessible for read and write operations by all nodes within a Windows Server Failover cluster.
- **PassThrough disk** can bypass the hypervisor layer and maximize the performance of the underlying disks.
- **NPIV** (N_Port ID Virtualization) is a Hyper-V virtual Fibre Channel feature introduced in Windows 2012. It allows connection to Fibre Channel storage from within a virtual machine. With NPIV, virtualized workloads can use existing

Fibre Channel investments. It also supports many related features, such as virtual SANs, live migration, and Multipath I/O (MPIO).

This feature provides close to pass-through disk performance with Hyper-V level protection and easy migration.

Virtual Fibre Channel for Hyper-V provides the guest operating system with unmediated access to a SAN by using a standard World Wide Name (WWN) associated with a virtual machine. Hyper-V users can now use Fibre Channel SANs to virtualize workloads that require direct access to SAN LUNs. Fibre Channel SANs also allow you to operate in new scenarios, such as running the Failover Clustering feature inside the guest operating system of a virtual machine connected to shared Fibre Channel storage.

EMC storage arrays are capable of advanced storage functionality that helps offload certain management tasks from the hosts to the SANs. Virtual Fibre Channel presents an alternate hardware-based I/O path to the Windows software virtual hard disk stack. This allows you to use advanced functionality like hardware snapshots directly from Hyper-V virtual machines. For details about Hyper-V virtual Fibre Channel, refer to the TechNet topic Hyper-V Virtual Fibre Channel Overview.

All-flash array storage considerations

EMC XtremIO

The EMC XtremIO all-flash array maximizes the use of flash storage media. Key attributes of the XtremIO platform are:

- High levels of I/O performance,
- Consistently low (sub-millisecond) latency
- True inline data reduction—removing redundant data in data path, write only unique data on storage array, to lower required capacity.
- A full suite of enterprise array capabilities, such as integration with VMware through vStorage APIs for Array Integration (VAAI), N-way active controllers (allows multiple active assignments), high availability, strong data protection, and thin provisioning.

EMC XtremIO best practices and considerations for SQL Server

SQL Server database file layout on XtremIO

When deploying SQL Server on XtremIO, many of the existing SQL Server flash storage considerations for traditional spinning hard disk media are no longer needed. XtremIO is optimized for both random and sequential access, therefore storage provisioning for SQL Server can be significantly simplified. With XtremIO, database administrators do not need to worry about RAID, spindle counts for performance purposes, or any of the storage configurations for SQL Server database best practice. The conventional SQL Server data file layout considerations are:

- RAID configuration
- Separation of SQL Server files
- Number of file groups / LUNs

- Number of files

RAID configuration

RAID configuration consideration is no longer relevant for XtremIO, because XtremIO ships with preconfigured flash optimized RAID.

XtremIO has a built-in "self-healing" double-parity RAID as part of its architecture. XtremIO Data Protection (XDP) takes advantage of flash media specific properties and XtremIO content addressable storage architecture. With the content addressable storage architecture, SQL Server files are automatically distributed across all SSDs, and processed by all storage processors in the cluster.

Separation of files

The common SQL Server recommendations are no longer required in XtremIO for performance reasons. On XtremIO, it is not necessary to allocate separate LUNs for data, log, and tempDB files or to allocate separate physical disks for LUNs used by data and log files. Also, allocating separate LUNs for different workload types (for example, OLTP versus Analytics) is not required.

In OLTP databases, SQL Server data file access is mostly random, while transaction log file access is sequential only. Traditional storage with spinning disk media requires repositioning of the disk head for random read and write access, making sequential data more efficient than random data access. Separating files with random from sequential access patterns helps to minimize disk head movements, thus optimizing storage performance.

XtremIO with all SSDs does not have disk head movement for read/write access, data is evenly distributed across all SSDs to benefit from the processing power of all storage processors. There is no performance impact in file separation for data, log, tempDB, or for different workload types.

From an operational efficiency or backup/recovery isolation perspective, this separation might still be used for overall SQL Server design.

You can follow best practice for separation of files in general for administration and operations, but not for performance considerations.

Figure 9 shows a sample configuration for a SQL Server instance in a virtualized environment. For a single database, the database LUNs are created on the same XtremIO LUN (same datastore) with log LUN, and configured as separate VMDKs. Depending on the need for better administration of snapshots and backup restore plan, the user can choose to separate data LUN and log LUN into different datastores. The tempdb however, has all the database files and the log file on the same datastore.

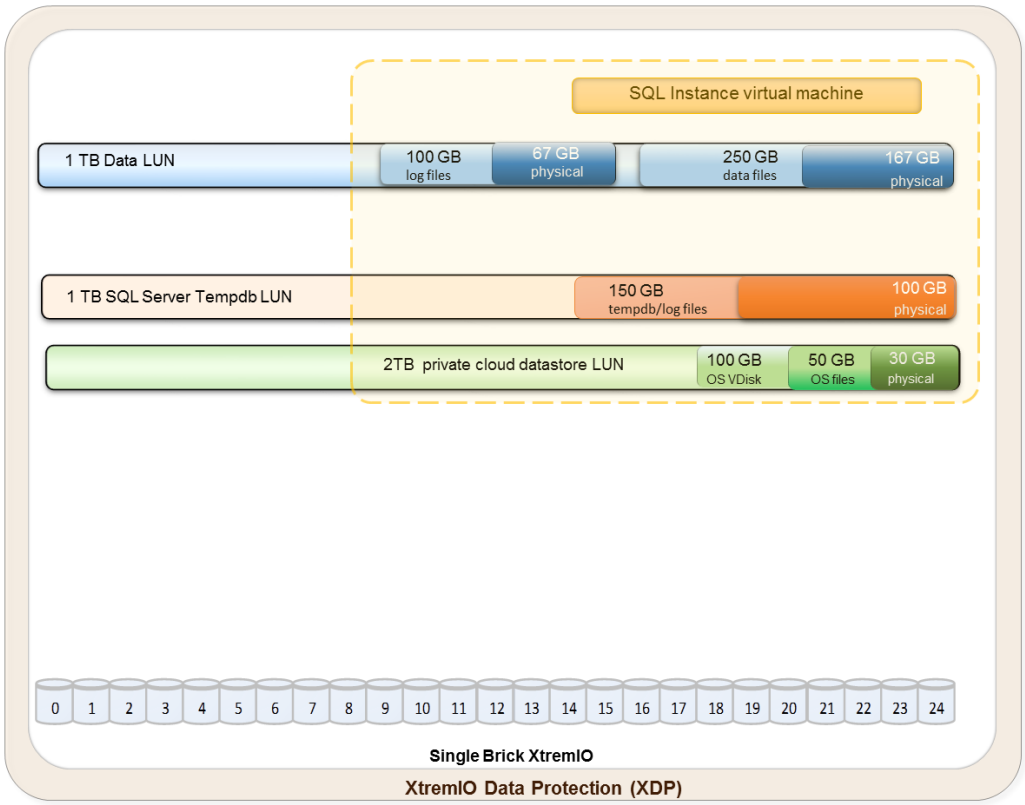

Figure 9. Sample database LUN configuration

Number of file groups/LUNs

The common considerations for SQL Server file groups are valid for XtremIO. File groups are used to separate user objects from one another into different LUNs or physical disks. SQL Server storage best practices recommend:

- Use file groups for administration requirements, for example, for backup and restore and partial database availability.
- Use data files to "stripe" the database across your specific I/O configuration, for example, physical disks and LUNs.
- Avoid selectively placing objects on separate LUNs to over optimize the I/O.

Continue to use multiple file groups to enhance database availability, manage table partitioning, and so on.

In general, to take full advantage of the parallel-processing powers of the multiple storage controllers in the cluster, and to maximize performance, consider the information in Table 13.

Table 13. Number of LUNs for data files of a single database needed to achieve maximum performance

Database type	OLTP	OLAP
Number of LUNs can be created for database files to achieve maximum performance	4	8

The best practice for numbers of file groups and LUNs:

- The number of file groups must still be created as common best practice.
- The number of LUNs for most of databases can be reduced to one or two.
- For maximum performance of OLTP, the number of data LUNs must be 4.
- For maximum performance of an OLAP database, the number of LUNs must be 8.

Note: Assume the maximum performance for OLTP IOPS >= 20,000, and OLAP bandwidth >= 1 GB/s for a single database.

Number of files

The recommendation for XtremIO follows the common recommendation of number for the files on a SQL Server database.

SQL Server "stripes" allocations across files within a file group by using a proportional fill algorithm. Each file has its own PFS, GAM, and SGAM pages. These special "administration pages" track the free space and allocation in the file. If the files in the group have the same size, which is recommended by SQL Server, the allocation is round-robin. Multiple files must be created to optimize database data files in a file group to improve workload with heavy insert activities, such as tempdb.

For details on best practice for the number of files needed for SQL Server databases, refer to the Microsoft support topic *Recommendations to reduce allocation contention in SQL Server tempdb database*.

The best practice for the number of files created for SQL Server database on XtremIO must follow the common practice.

Deployment considerations

Multipathing

The best practices for multipathing are in Table 14:

Table 14. Multipathing best practices

Application	Best practice
Windows Server	MPIO policy should be set as “Least Queue Depth”
VMware vSphere	On ESXi server, set “round robin” as multipath policy
	Set NMP Round Robin path switching frequency to 1(instead of the default 1000)
PowerPath	https://support.emc.com/docu56210_XtremIO-2.2.x---4.0-Host-Configuration-Guide.pdf?language=en_US

Physical sector size

XtremIO LUNs can be created with either **512B** or **4K** physical sector size for SQL Server.

Microsoft started native support for 4K sector size with release Windows 2012.

4K sector size aligns with SQL Server 8 KB data page boundary efficiently, reducing the amount of metadata overhead. With ODX copy (which is a heavy metadata operation) of 1 TB SQL Server data, the average bandwidth can be doubled on 4K sector size volumes compared with that of the 512B sector volume.

An XtremIO LUN created with 4K sector size has some limitations:

- SQL Server transaction log writes with a large number of small write operations with frequent commits may see an increased use of log space.

  Note: SQL Server write block always aligns with the physical sector size. Log write block size can be one or a multiple of the physical sector size with up to 60Ks.

- It may not be supported for the versions of SQL Server or Hypervisor.

The best practices for determining and configuring a proper sector size for volumes are:

- Before deploying a SQL Server database, always check on the physical sector size of the volumes that are provisioned for SQL Server by running FSUTIL fsinfo ntfsinfo ‹drive› on Windows. Ensure all volumes for SQL Server databases, including system and tempdb, are on the volumes with the same sector size.

- When moving an existing database to the XtremIO platform, check the sector size on the existing database using dbcc fileheader. Align the XtremIO volume sector size with the sector size of the existing database.

- Use 4K sector size for physical Windows and Hyper-V deployments with standalone XtremIO deployment.

- Use 512B sector size if the deployment includes products such as EMC RecoverPoint™ for replication, or EMC VPLEX®.
- Use 512B sector size for SQL Server on VMware deployments.

Database native compression considerations on XtremIO

Row and page compression

You can enable SQL Server compression at the **row** or **page** level for rowstore tables and indexes. SQL Server also supports **columnstore** compression for columnar tables and indexes:

- **Row compression** changes the format of the data type declared for a column from fixed length to variable length.
- **Page compression** implements row compression and uses prefix and dictionary compression algorithms to compact data within pages.

Before enabling SQL Server compression, evaluate the impact on database performance. The database server requires additional CPU resources because pages are compressed and decompressed in memory. However, XtremIO offers both compression and deduplication technology which is always enabled with no measurable impact on database performance.

SQL Server native compression aims to efficiently store data by performing deduplication and compressing it within a single row, page, or column segment. For XtremIO, deduplication and compression works across **all database pages** for the entire array. Both SQL Server and storage-side compression technologies may offer gains and complement each other on the data in your environment.

In a series of tests, we measured the space savings benefits and overall performance impact of using the native capabilities of SQL Server row and page compression for rowstore tables and indexes, as shown in Figure 10. We also measured its impact on the host CPU utilization and the impact on the volumes managed by XtremIO

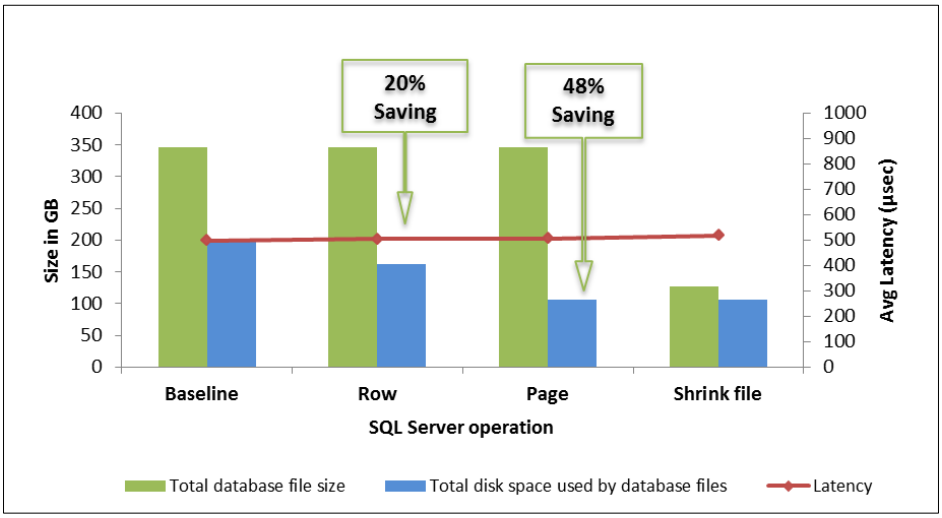

Figure 10. Space savings from SQL Server native compression

SQL Server compression and decompression operations increase CPU utilization at the host level, as shown in Figure 11.

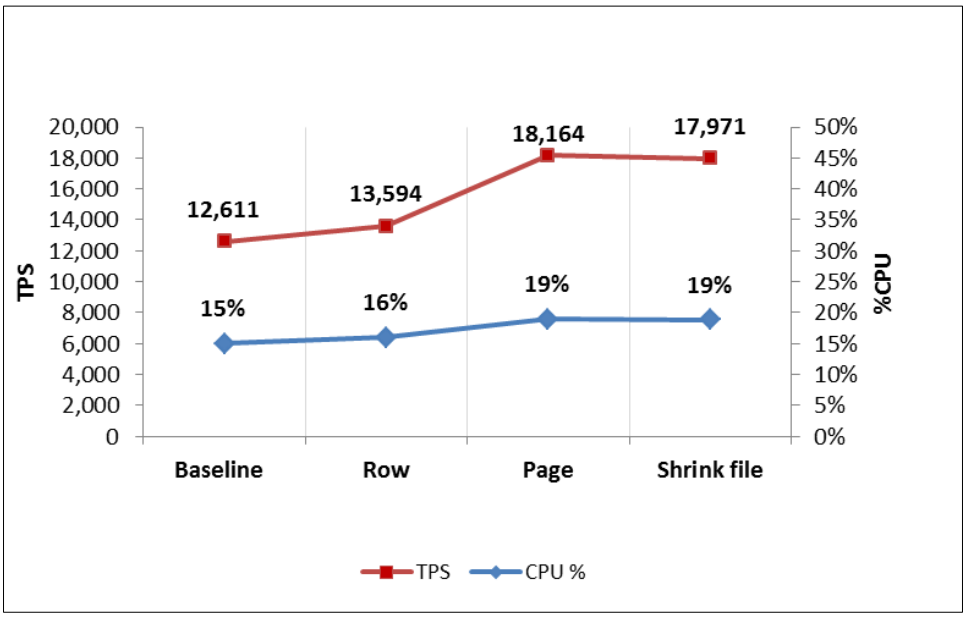

Figure 11. CPU consumption from SQL Server native compression

Note: The result of SQL Server native compression benefits, shown in Figure 11, is based on a test environment for a single 250 GB database, which may vary based on the actual SQL Server database environment.

Columnstore compression

With columnstore compression, data pages contain date from a single column rather than row(s) from an index. Data is automatically compressed across a segment. A segment can span multiple pages. I/O requests are in segments not pages.

Since SQL Server 2012, for large data warehouse queries, columnstore index can offer improved processing times. SQL Server 2012 supports non-clustered columnstore index, the underlying base table has to be of rowstore structure. Additionally, in SQL Server 2012 the columnstore index is read-only. SQL Server 2014 supports updateable columnstore indexes and the base table may be columnar. This increases the usage scenarios and space savings through compression.

Columnstore is a memory-optimized technology. In Figure 12, we compared processing of rowstore and columnstore index for the same data and query set.

For large data warehouse queries, columnstore results in decreased I/O and increased CPU usage.

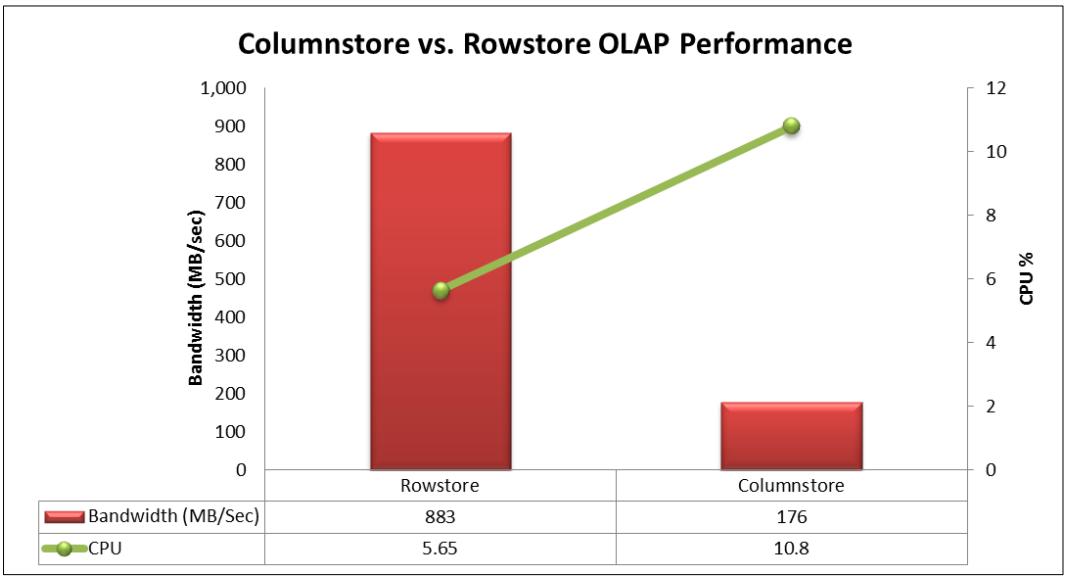

Figure 12. CPU and bandwidth usage for Columnstore and Rowstore processing

For large data warehouse queries, in-memory columnstore index can offer improved processing times, as shown in Figure 13.

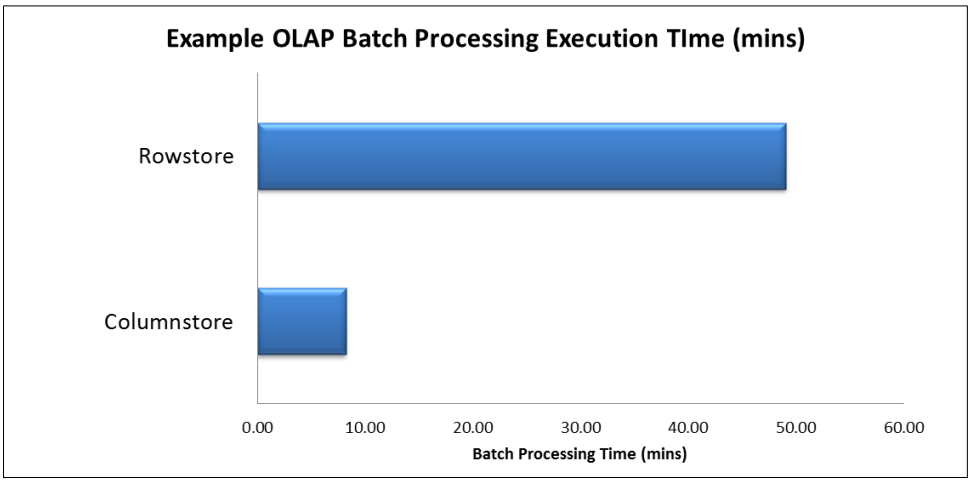

Figure 13. OLAP batch processing time for Rowstore compared to Columnstore

Database encryption considerations on XtremIO

SQL Server Transparent Data Encryption

Transparent Data Encryption (TDE) is the native SQL Server encryption at rest technology which encrypts the database data and log files. When implemented on any database on a SQL Server instance, the SQL Server system database tempdb is also encrypted. SQL Server data pages are encrypted or decrypted when they are transferred to and from disk and remain decrypted in memory, therefore the more I/O dependent the workload the greater the impact on performance. TDE may also impact storage capacity by eliminating the benefits from array based compression and deduplication.

Data At Rest Encryption

The XtremIO all-flash array uses high performance inline encryption to ensure all the data stored on SSDs is encrypted at rest. DARE is implemented by default on new installations of XtremIO version 4 and can be turned on in previous versions. DARE offers superior performance over TDE, offloads encryption processing to the array, and does not have the physical capacity limitations associated with TDE.

Considerations for using TDE

If using TDE on XtremIO, features such as array compression and thin provisioning, which save space, are not available.

Any snapshots taken before TDE is implemented continue to be zero optimized (zero disk for pre-allocated space) and the array compressed. Because the data is completed differently from the TDE copy, the snapshot copy no longer shares physical data blocks with the source volume.

New snapshots taken after encryption benefit from XtremIO space saving even for encrypted DB.

Consider using a combination of XtremIO DARE and SQL Server (2014 only) or EMC Avamar® backup encryption to ensure the database at rest is encrypted at all times. Figure 14 shows additional capacity requirements for SQL Server database using both XtremIO DARE and SQL Server TDE.

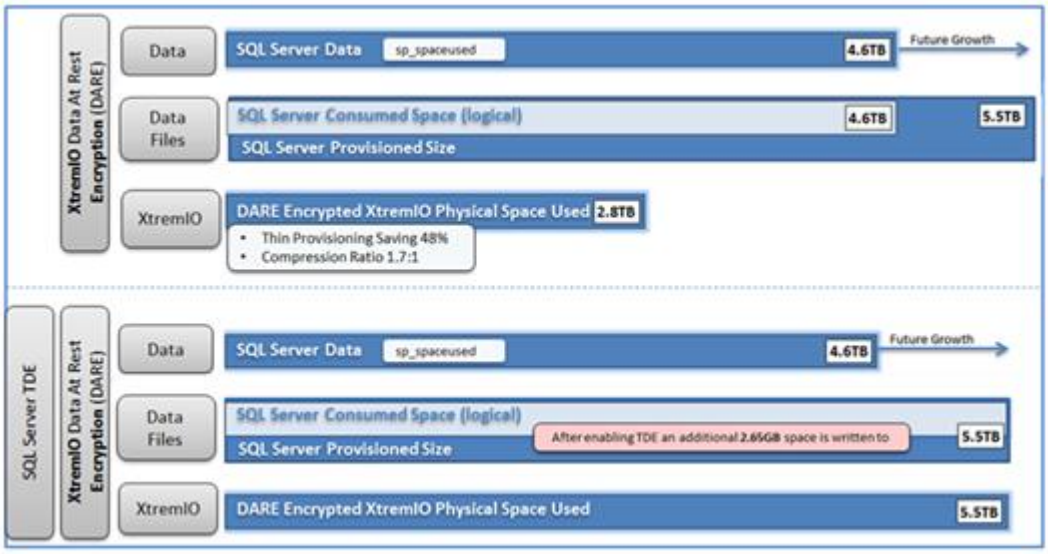

Figure 14. Capacity requirements for SQL Server data using XtremIO DARE versus SQL Server TDE

EMC VMAX storage design guidelines

The EMC VMAX and VMAX3 families of storage arrays deliver the latest in Tier-1 scale-out multi-controller architecture with consolidation and efficiency for the enterprise. The EMC VMAX3 family of storage arrays can scale up to eight engines and consolidates more workloads with a much smaller footprint than other arrays.

The VMAX3 Dynamic Virtual Matrix architecture enables IT departments to build storage systems that transcend the physical constraint of competing array architectures. The HYPERMAX OS is the industry's first open-storage and hypervisor-converged operating system, which combines high availability, I/O management, quality of service, data integrity validation, storage tiering, and data security with an open application platform.

The EMC VMAX3 family also offers increases in floor-tile density with engines and high-capacity disk enclosures for both 2.5-inch and 3.5-inch drives that are consolidated in the same system bay. These VMAX3 models come fully preconfigured from the factory to significantly shorten the time from installation to first I/O.

VMAX3 - SLO considerations and best practices

The EMC VMAX3 family introduced the Service Level Objective (SLO) feature that enables building different levels of service and assigning them to the specific workloads to meet performance and availability requirements. SLO fits various SQL Server workloads and can dynamically adjust the service level for different use cases. Table 15 lists the general recommendations of SLO selection for SQL Server workloads.

Table 15. General recommendation of SLO selection

SLO	Performance type	Use case
Diamond	Ultra high	High-performance computing, latency sensitive
Platinum	Very high	Mission-critical, high rate OLTP
Gold	High	Heavy I/O, database logs, data sets
Silver	Price/performance	Database datasets, virtual applications
Bronze	Cost optimized	Backup, archive, file
Optimized (default)	N/A	Places the most active dataset on the highest-performing storage, and the least active on the most cost-effective storage

SLO considerations for data files

The EMC VMAX3 family introduces new levels of simplicity and automation to make storage provisioning and allocation easy and fast. You do not need to be familiar with SQL Server database design to achieve high performance and availability.

You can configure VMAX3 as an all-flash array or as a hybrid with multiple storage tiers. The choice of SLO is irrelevant with an all-flash array. For a hybrid VMAX3 configuration, the default SLO can be used for simplicity, providing best performance and availability across the whole storage system. However, VMAX3 also offers users

finer control over database performance if the business wants to provide more defined performance guidelines to certain SQL Server databases. This can be done by specifying a distinct SLO and workload type, which together maintain the application response time at a predictable range and under user requirements.

VMAX3 offers four types of workload types when designing your SQL Server databases:

- **OLTP**: Recommended for transactional application workloads, typically for small I/O operations
- **DSS**: Recommended for data warehousing application workloads, typically sequential read/write with large I/O operations
- **OLTP + Replication**: Recommended for OLTP workload types in Symmetrix Remote Data Facility (SRDF®) use cases
- **DSS + Replication**: Recommended for DSS workload types in SRDF use cases

VMAX3 SLO also allows the user application to be kept in a predefined response time compliance range so that the application behavior can be predictable. When selecting the SLO, EMC recommends that you select the best SLO definition based on the application response time requirements as listed in Table 16. For details, refer to *Managing Microsoft SQL Server Workloads by Service Levels on EMC VMAX3 Solution Guide*.

Table 16. SLO definitions and expected response times

Service Level Objective	Behavior	Expected average response time for small I/O (OLTP)	Compliance range (OLTP)	Expected average response time for large I/O (DSS)	Compliance range (DSS)
Diamond	Emulates flash drive performance	0.8 ms	0-3 ms	2.3 ms	1–6 ms
Platinum	Emulates performance between flash drive and 15k RPM drive	3.0 ms	2-7 ms	4.4 ms	3–9 ms
Gold	Emulates 15k RPM performance	5.0 ms	3-10 ms	6.5 ms	4–12 ms
Silver	Emulates 10k RPM performance	8.0 ms	6-15 ms	9.5 ms	7–17 ms
Bronze	Emulates 7.2k RPM NL-SAS drive performance	14.0 ms	10-25 ms	15.5 ms	11-27 ms

SLO considerations for tempdb and log files

SQL Server log files typically exhibit sequential write I/O behavior. Because all writes in VMAX3 go to cache first, the SLO selection has a limited effect on the log performance. Therefore, SQL Server log files can be put on any SLO as required. EMC recommends selecting the Optimized (default) SLO for simplicity or the same SLO as is used for the data files. Particularly, in cases where log latency is sensitive, select Platinum or Diamond SLO as the best storage tiers for SQL Server log files.

SQL Server tempdb can exhibit random or sequential I/O behavior with various sizes, depending on the application workloads that are running together. EMC recommends that you separate tempdb into multiple files and put them on the Optimized (default) SLO level to fit different workload demands. For the uses cases that tempdb accesses excessively, such as high performance OLTP, or frequent reporting containing large table joins, you can select Platinum or Diamond SLO to service the tempdb high-activity rate.

For more details, refer to white paper *Deployment Best Practices for Microsoft SQL Server with VMAX3 SLO Management*.

Pre-VMAX3 considerations

VMAX series hardware design considerations

Some of the most important design considerations for SQL Server on VMAX are:

- When creating LUNs, use fewer but larger hyper volumes to improve performance.
- Use a minimum of two HBAs per server, with each HBA connected to at least two director ports (across multiple VMAX engines, if possible).
- For thick and thin LUNs, use striped metavolumes.

Virtual provisioning considerations and best practices

With Microsoft Windows “thin-friendly” NTFS formatting capabilities and Microsoft SQL Server’s Instant File Initialization mechanisms, SQL Server can derive maximum benefit from virtual provisioning with EMC Symmetrix. Thus, thin LUN pools are recommended for SQL Server on Symmetrix VMAX. Thin device performance is equivalent to regular (thick) device performance on VMAX, and in most cases, the use of thin pools can reduce the initial storage requirement.

The following summarizes best practices when configuring Microsoft SQL Server with virtual provisioning on EMC Symmetrix VMAX:

- Use virtual provisioning when system over-allocation of storage is typical.
- Use virtual provisioning when expecting rapid growth over time but downtime is limited.
- Configure SQL Server databases with Instant File Initialization (this is the default for SQL Server 2012).
- Consider using the thin pool utilization threshold tool to monitor the pools and avoid thin pools running out of space.

Avoid virtual provisioning for the following environments:

- Systems where shared allocations from a common pool of thin devices do not meet the customer requirements.
- Systems where a large amount of deleted space cannot be reclaimed.
- Systems that cannot tolerate an occasional response time increase of approximately 1 millisecond because of writes to uninitialized blocks.

FAST VP considerations and best practices for a VMAX storage system

FAST VP provides SQL Server with reduced administration and faster space and I/O issue resolution. Especially with OLTP workload, FAST VP provides efficient storage usage and moves the most frequently accessed data to the highest performing tier.

The following are some considerations and best practices when using FAST VP:

- Avoid putting log LUNs in a flash tier because logs generate mostly sequential writes and will not benefit from a flash device.
- Bind database LUNs to the FC tier to start with FAST VP and provide enough storage to accommodate the capacity if the hot data needs to move to the upper tier.
- When using FAST VP with AlwaysOn Availability Groups, place availability group copies of the same database in different pools to achieve better availability.
- When considering sizing on VMAX using FAST VP, note there can be up to three tiers. With an OLTP workload, assume 75/15/10 of hot/warm/cold skew for I/O, then adjust according to the actual environment:
    - 10 percent I/O and 75 percent capacity on SATA
    - 15 percent I/O and 15 percent capacity on FC
    - 75 percent I/O and 10 percent capacity on flash

  Note: For details about sizing for FAST VP tiering, refer to the *EMC Virtual Infrastructure for Microsoft Applications enabled by Symmetrix VMAX and Microsoft Hyper-V* White Paper.

- Logs can be pinned in a specific tier when placed in the same storage pool with data file LUNs.

Note: The skew ratios can vary and depend on the specific SQL Server profile.

EMC VNX storage design guidelines

The EMC VNX family delivers industry-leading innovation and enterprise capabilities for file and block storage in a scalable, easy-to-use solution. This next-generation storage platform combines powerful and flexible hardware with advanced efficiency, management, and protection software to meet the demanding needs of today's enterprises.

The VNX family includes the VNXe series, purpose-built for the IT manager in entry-level environments, and the VNX series, designed to meet the high-performance, high-scalability requirements of midsize and large enterprises.

EMC FAST Suite is an advanced software feature that provides greater flexibility to manage the increased performance and capacity requirements of the SQL Server environment. EMC FAST Suite makes use of SSD drives, SAS, and NL-SAS storage configuration to balance performance and storage needs. FAST Suite includes FAST Cache and FAST VP.

Application Protection Suite automates application-consistent copies and enables you to recover to defined service levels. User roles enable self-service copy management, while improving visibility for all application recovery points. Alerts are generated automatically, providing fast resolution to recovery gaps. Integrated reporting can prove compliance with protection policies. Applications supported include Oracle; SQL Server, and SharePoint; VMware; and Hyper-V. Application Protection Suite includes:

- For VNX series: Replication Manager, AppSync®, and Data Protection Advisor™ for Replication Analysis
- For VNXe series: Replication Manager

EMC FAST Cache best practices and considerations

EMC FAST Cache increases the storage system cache by extending the functionality of DRAM cache, mapping frequently accessed data to SSD. FAST Cache capacities range from 73 GB to 2 TB, which is considerably larger than the available DRAM cache of the existing storage systems. If the user application accesses a particular chunk of data frequently, that chunk is automatically promoted into the FAST Cache by copying it from hard disk drives to the flash drives. Subsequent access to the same chunk is serviced at flash drive response times, thus boosting the performance of the storage system.

FAST Cache is most suitable for I/O-intensive random workloads with small working sets. A typical OLTP database with this kind of profile can greatly benefit from FAST Cache to improve performance and response time. Monitor SQL Server database file groups and enable FAST Cache on the highly active storage pools where such data is located.

The *EMC FAST Cache: a Detailed Review* document, available on the EMC Online Support website, provides more details on FAST Cache design concepts, planning, and usage guidelines.

Testing suggests that the inclusion of FAST Cache results in a 300 percent increase in transactions per second (TPS) for a SQL Server OLTP workload using the same number of hard disk drives in the back end.

The *EMC Unified Storage for Microsoft SQL Server 2008: Enabled by EMC CLARiiON and EMC FAST Cache Reference Architecture Guide* available on the EMC Online Support website provides more details on how to build a solution with EMC FAST Cache.

When using FAST Cache, allow sufficient time for the cache to warm up to fully utilize the cache. In our tests with OLTP workloads, FAST Cache took about 1 to 2 hours to warm up.

The warm-up time depends on the type and number of back-end hard disk drives, the size of FAST Cache, and the size of the working set. EMC Unisphere has several FAST Cache counters that you can monitor to obtain optimum utilization and warm-up time. The *EMC CLARiiON, Celerra Unified, and VNX FAST Cache White Paper*, available on EMC.com, provides more information.

EMC FAST VP best practices and considerations

According to industry analysts, 60 to 80 percent of operational database data is inactive and increases as the size of the database grows. Low-cost, high-capacity spinning drives are an ideal choice for inactive data, while high performance drives are suitable for frequently accessed data. Manually classifying and storing data in the right tier is a complex and challenging task, and usually requires down time to move data.

EMC FAST VP automatically moves frequently accessed data to a faster physical storage within a pool and moves infrequently accessed data to a less-expensive physical storage within the pool.

Using FAST VP you can:

- Control when FAST VP can move data to prevent any impact on host I/O requests during known periods of high system usage.
- Improve overall system performance without investing in additional high-performance physical drives.
- Apply FAST VP to any or all pool-based database LUNs on a storage system.
- When log LUNS need to be in the same pool of database LUNs, pin them to the SAS tier and disallow data relocation for these LUNs.
- In an OLTP workload, pin the tempdb in the SAS tier and disallow data relocation for these LUNs.
- Set the FAST policy to Start High then Auto Tier (default).

Understanding Microsoft SQL Server OLTP Performance Characterization for EMC VNX Series, available on the EMC Online Support website, provides more details about using FAST VP with SQL Server.

FAST VP operates in the background, while the LUNs are online and available for host access. You can control the data movement speed to minimize the impact on overall system performance (you can configure the relocation rate to high, medium, or low).

Best practices for FAST VP sizing

A small percentage of the total utilized capacity is typically responsible for most of the I/O activity in a given database. This is known as workload skew. Analysis of an I/O profile may indicate that 85 percent of I/Os to a volume only involve 15 percent of the capacity. The resulting active capacity is called the **working set**. Software such as FAST VP and FAST Cache can keep the working set on the highest-performing tier.

The following are some best practices for sizing FAST VP:

- The working set must be in the highest tier (FC/SAS/flash). A typical working set for an OLTP workload is 10 to 30 percent of the database file. Size the top tier in FAST VP to ensure that the working set can be migrated to the highest performance tier.
- Because FAST VP Performance Tier drives are versatile in handling a wide spectrum of I/O profiles, EMC recommends having Performance Tier drives in each pool.
- High capacity drives can optimize TCO and often compose 60 to 80 percent of a pool's capacity. Profiles with low IOPS/GB or sequential workloads can use a larger number of high capacity drives in a lower tier.
- I/O skew is important to determine prior to sizing FAST VP tiers. SQL Server skew can vary and depends on the actual SQL Server profile.
- For most VNX sizing tasks, use two tiers only: a performance tier (FC/SAS/flash) and a capacity tier (SATA/NL-SAS). Assume an OLTP workload of 85/15 skew, and adjust according to the actual environment:
  - 85 percent I/O with 15 percent capacity on performance tier (FC/SAS/flash)
  - 15 percent I/O with 85 percent data on capacity tier (SATA/NL-SAS)
- EMC Professional Services and qualified partners can assist you to size tiers and pools properly to maximize investment. They have the tools and expertise to make specific recommendations for tier composition based on an existing I/O profile.

Sizing for performance

The IOPS and disk calculation is the same as that given for the RAID group with skew:

$$\text{Number of Disks for tier} = \frac{Read\ IOPS + (Write\ IOPS \times RAID\ performance\ overhead)}{IOPS\ per\ disk} \times I/O\ skew\ for\ tier$$

Note: Each tier needs to round up to the next logical drive count as the tier with its specific RAID type. For example, for RAID 5 (4+1), a 10- disk set, instead of a seven-disk set, is required.

Capacity-based calculation

Calculate the disk spindles based on the capacity for each tier with the following formula:

$$Number\ of\ disks\ for\ tier = \frac{Total\ capacity \times capacity\ skew\ for\ tier}{RAID\ capacity\ overhead}$$

Final design

Use the larger number for each tier as the final sizing to satisfy both performance and capacity requirements.

FAST Cache compared to FAST VP

FAST Cache boosts the performance immediately for random-access patterns and burst prone data, while FAST VP allows the storage system to move inactive data to a lower-cost storage tier. FAST Cache operates in 64 KB chunks of storage. FAST VP can promote both thin metadata and the most-accessed user data to the higher tier, while FAST Cache promotes the metadata to improve thin LUN performance.

FAST Cache and FAST VP can work together in a SQL Server database to improve performance, achieve higher capacity utilization, and lower the power and cooling requirements.

The following are best practices when using FAST Cache and FAST VP:

- When a limited number of flash drives are available, use flash drives to create FAST Cache first.
  - FAST Cache is global and can benefit multiple pools in the storage system.
  - FAST Cache has higher granularity (uses 64 KB chunks) than FAST VP (1 GB chunks), which results in higher performance benefits and faster reaction time for changing usage patterns.
- Use flash drives to create a FAST VP performance tier for a specific pool to ensure the performance of mission-critical data. FAST VP tier is dedicated to a storage pool and is not shared with other storage pools in the same storage array.

DARE impact and considerations

The VNX DARE feature helps to protect SQL Server database data against physical theft or loss. It uses a controller based encryption method to encrypt all data written to disk, protecting data access against unauthorized drive removal. DARE encrypts the entire array at the drive level. This controller based encryption technology has minimal or close to zero performance impacts on SQL Server workloads. The encryption operation is fully offloaded to the hardware encryption module. EMC recommends enabling the DARE feature on the VNX array while running the SQL Server database application that has high security requirements.

Software-based storage considerations

EMC ScaleIO

ScaleIO is a software-only solution that uses existing servers' local disks and LAN to create a virtual SAN that has all the benefits of external storage—but at a fraction of the cost and complexity. ScaleIO turns the existing local internal storage into internal shared block storage. For many workloads, ScaleIO storage is comparable to, or better than, external shared block storage.

The ScaleIO software components are installed on the application servers and communicate via a standard local area network (LAN) to handle the application I/O requests sent to ScaleIO block volumes. An efficient decentralized block I/O flow, combined with a distributed, sliced volume layout, results in a massively parallel I/O system that can scale up to thousands of nodes.

The ScaleIO virtual SAN consists of the following software components:

- **Meta Data Manager (MDM):** Configures and monitors the ScaleIO system. The MDM can be configured in single mode on a single server, or in redundant cluster mode, with three members on three servers.
- **ScaleIO Data Server (SDS):** Manages the capacity of a single server and acts as a back end for data access. The SDS is installed on all servers contributing storage devices to the ScaleIO system. These devices are accessed through the SDS.
- **ScaleIO Data Client (SDC):** A device driver that exposes ScaleIO volumes as block devices to the application, residing on the same server on which the SDC is installed.

Hardware, pools, MDM, and SDS considerations

Host connectivity

The ScaleIO software components communicate via a standard LAN to handle the application I/O requests sent to ScaleIO block volumes. An efficient decentralized block I/O flow, combined with a distributed, sliced volume layout, results in a massively parallel I/O system that can scale up to thousands of nodes.

EMC suggests using two 10 GbE networks dedicated to ScaleIO data traffic.

Figure 15 shows an example network topology. In the example, the two 10 GbE networks were dedicated to ScaleIO data traffic, and were separated from 1 GbE application networks.

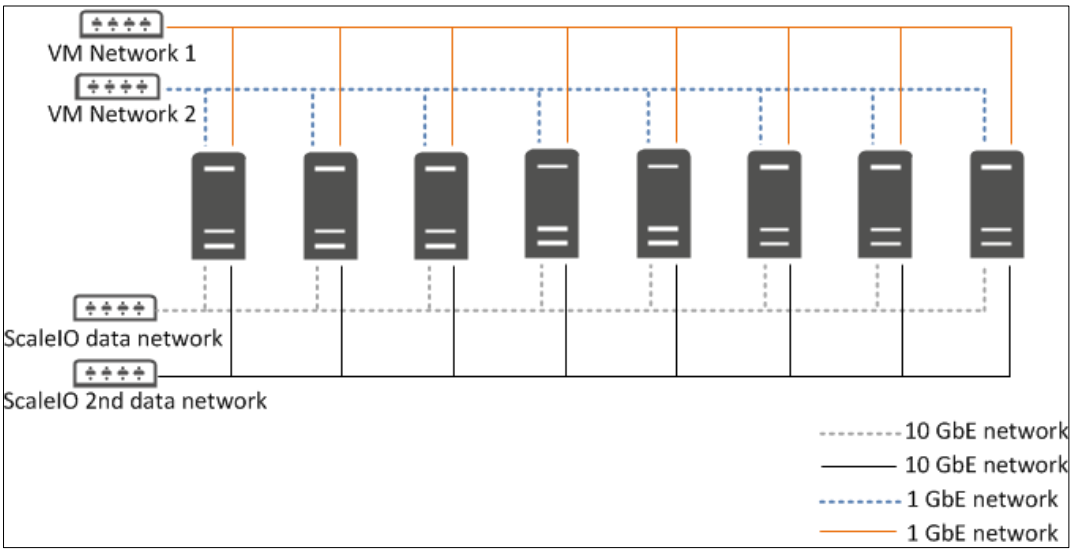

Figure 15. Network topology example

DAS configuration

ScaleIO works with any free capacity—internal or direct-attached devices, either HDD or flash-based devices such as SSD and PCIe cards. Although ScaleIO works with any device topology, EMC recommends that you configure the raw devices as stand-alone devices.

If the server has a RAID controller, ScaleIO prefers to use the controller's caching abilities for better performance, but is better utilized when all devices are configured as stand-alone (for example, setting each of the devices to RAID 0 separately). For HDD devices, EMC recommends that you enable RAID-controller caching. Use read/write caching if the cache is battery-backed, or use **read only** caching if the cache is not battery-backed. For flash devices, the caching method used depends on the device behavior.

Disk layout and storage pool configuration

After the physical layer is ready, the ScaleIO MDM cluster aggregates all storage exposed to it by all of the SDSs to generate a virtual layer, that is, virtual SAN storage. You can define volumes over the storage pools and expose them to the applications as a local storage device using the SDCs.

ScaleIO can benefit from massive I/O parallelism when you distribute the storage pool to as many nodes and disks as possible. Therefore, when there are multiple applications each requiring a small storage pool, usually you can get optimal performance by combining these applications in a single storage pool on ScaleIO. Moreover, even with the same amount of disks, it is a best practice to distribute the pool to more nodes when possible. For example, a storage pool with eight nodes (each contributing two disks) is preferable to a storage pool with four nodes (each contributing four disks), although both storage pools contain 16 disks.

The storage design must also meet the application requirements and business requirements. For standard database applications with built-in high availability features, such as SQL AlwaysOn AG, the active copy of data must never be stored on the same physical drives as the passive copy to avoid a single point of failure. In

addition, the business may require a certain level of isolation so that a malfunctioned application does not affect other applications. This may not be guaranteed when these applications share the same storage pool.

SQL Server recommendations

For availability, performance, and flexibility, ensure optimal storage sizing and virtual machine configuration for SQL Server.

ScaleIO supports HDDs, SSDs, and PCIe flash cards to create a virtual pool of block storage with varying performance tiers. Choose the appropriate disk type for SQL Server data and log pools based on the workload. Also, assign a sufficient number of disks to the pools to ensure the ScaleIO system can handle I/O spikes smoothly. The ScaleIO scalability of performance is linear with regard to the growth of the storage device (nodes and local storage devices). Throughput and IOPS scale in direct proportion to the number of servers and local storage devices that are added to the system.

Note: The following EMC white paper *EMC Converged Infrastructure for Microsoft Applications* introduced an example deployment of SQL Server on ScaleIO.

SQL Server best practices suggest that tempdb, data, and log files must be separated on different LUNs to achieve high performance and the ability to restore a database when data corruption occurs. To further enhance the high availability of SQL Server, use the AlwaysOn Availability Groups feature. As a best practice, we suggest separate availability replicas into different pools on ScaleIO.

Note: The tempdb in typical OLTP environments generally produces a small number of semi-random IOPS, while the tempdb log file produces minimal small sequential writes. The database log file produces small highly sequential IOPS, almost exclusively writes.

For industry-standard modern OLTP workloads on ScaleIO, it is a best practice to set the database checkpoint setting to **Automatic** or **Indirect**.

Note: Automatic checkpoints are throttled based on the number of outstanding writes and whether the Database Engine detects an increase in write latency above 20 milliseconds. Indirect checkpoints can reduce checkpoint-related I/O spiking by continually writing pages that are modified in the cache but not yet written to the disk (dirty pages) to a disk in the background.

Automation with ESI

EMC Storage Integrator (ESI) for Windows Suite is a set of tools for Microsoft Windows and Microsoft applications administrators. The suite includes ESI for Windows, ESI PowerShell Toolkit, ESI Service, ESI Management Packs for System Center Operations Manager (SCOM), and ESI Service PowerShell Toolkit.

ESI for Windows

Provides the ability to view, provision, monitor, and manage block and file storage for Microsoft Windows. ESI supports the EMC Symmetrix VMAX series and EMC VNX series. ESI also supports storage provisioning and discovery for Windows virtual machines running on Microsoft Hyper-V, Citrix XenServer, and VMware vSphere.

As shown in Figure 16, ESI simplifies the storage management for Windows and automatically applies some best practices with storage configuration thus simplifying storage deployment for Windows.

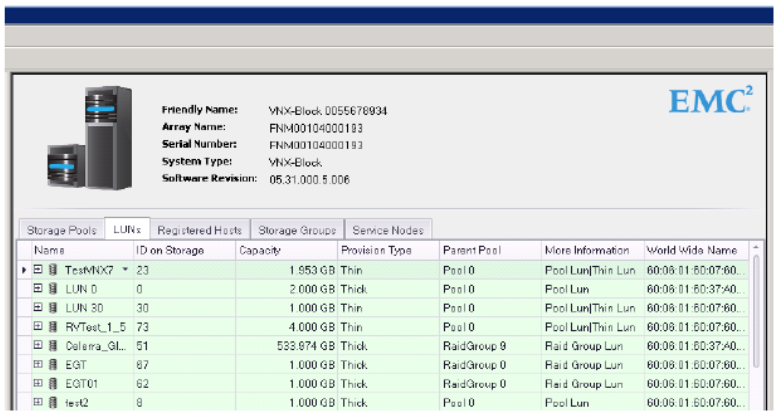

Figure 16. ESI for Windows storage management

ESI Management Packs for Systems Center Operations Manager 2012

Enable storage infrastructure management in a single user interface. With the SCOM 2012 integration, Windows administrators can discover storage assets, map physical objects and logical objects, manage alerts, and roll-up health states, with configurable parameter thresholds from within System Center 2012.

EMC PowerShell Toolkit (PSToolkit)

Is a powerful utility designed for administrators and users of Windows to assist in storage system management. PSToolkit cmdlets enable storage system administrators to get storage system information; create and delete storage pools, storage groups, and storage volumes; and map and mask these to available host servers. This toolkit enables administrators to efficiently create automated scripts for dynamic creation and deletion of virtual machines as needed by users.

Chapter 5 SQL Server Protection and Availability

This chapter presents the following topics:

Overview

Microsoft enhanced its native SQL Server high availability and data protection capabilities at the database level for SQL Server 2012 by introducing the AlwaysOn Availability Groups feature. EMC has a number of data protection products and options that complement AG and can further protect your SQL Server environment from the loss of a database, server, or an entire site. Various SQL Server 2012 high availability and disaster recovery options are described in this section.

EMC storage systems offer a wide range of features for SQL Server database protection and high availability. EMC replication technologies, such as TimeFinder®, SRDF and VNX snaps/clones, provide best data protection in the industry. RecoverPoint with continuous protection and Replication Management tools, such as AppSync and Replication Manager, provide protection for SQL Server in the application level.

You can use EMC technology in the SQL Server backup processes to:

- Reduce production system impact during the backup processing.
- Create consistent backup images.
- Integrate SQL Server backup and recovery processes.

Native data protection

If a past point-in-time copy of a database is required, you can use SQL Server to create a lagged copy in an AG environment. This can be useful if a logical corruption replicates across the databases in the AlwaysOn Availability Group, which results in the return to a previous point in time. This is also useful if an administrator accidentally deletes the user data. EMC has the ability to provide the same or better protection levels but use far less storage with the use of snapshots.

Recoverable versus restartable copies

EMC replication technologies can create two different types of database copies, recoverable or restartable. You can use either or both technologies to meet the needs of backup RPO and other requirements.

Recoverable database copy

A recoverable database copy is a backup in which logs can be applied to the database and rolls forward to any point in time after the database copy was created. In the event of a production database failure, the database can be recovered not only to the point in time of the last backup, but can also roll forward the subsequent transactions up to the point of the failure. This is a very important feature for the SQL Server database as well as for many other business requirements. Table 17 lists the three ways of creating a recoverable copy of a SQL Server database.

Table 17. SQL Server methodology to create a recoverable database copy

Backup methodology	Description	Supported
Stream backup	T-SQL statement or native SQL Server backup	Native SQL backup, Data Domain Boost, Networker®
VDI	Virtual Device Interface for third party software	VMAX/VNX/VNXe Networker, AppSync, RecoverPoint
VSS	Volume Shadow Copy Service for third-party software	VMAX/VNX/VNXe Networker, AppSync, RecoverPoint

All three types of backup are integrated with SQL Server and are considered to be hot backups. SQL Server records the backup when it takes place.

Restartable database copy

When a database copy is created in a storage level without any database-level integration, SQL Server can use the database copy to perform a crash recovery and bring the database to the point in time that copy was taken. This is considered a restartable database copy.

In this case, all transactions recorded as committed and written to the transaction log, with its corresponding data pages written to the data files, are rolled forward (redo). SQL Server will then undo or roll back changes recorded but never committed (such as dirty pages flushed by a lazy write).

This results in a database state with a transactionally consistent point in time. Additional transaction log backup cannot be applied to a database in this state, thus making the database recovered only to the point in time of the backup.

A restartable database copy is considered as a cold copy of the database. There is no record of backup in SQL Server.

EMC Consistency technology can be used to create restartable copies of a source SQL Server database. This type of technology is noninvasive to production database operations and occurs entirely at the storage array level. The restartable images represent dependent-write consistent images of all related objects that have been defined within the consistency group.

VDI and VSS frameworks for backup replication

EMC VMAX, VNX, and VNXe also implement consistency technology integrated with VDI snapshot technology and VSS framework to create a recoverable database copy.

EMC RecoverPoint and AppSync are built on top of these technologies to provide data protection to suit different environments.

EMC high availability and disaster recovery for SQL Server

While SQL Server native data protection is sufficient for some customers, most still require full backup and restore capabilities for SQL Server databases. EMC offers a wide range of options to provide high availability, disaster recovery and data protection with SQL Server, as listed in Table 18.

Table 18. EMC high availability, disaster recovery, and data protection options

Category	Tool/system	Features	Description
Continuous availability and disaster recovery	RecoverPoint	CDP	• Synchronous • Local recovery protection
		CRR	• Asynchronous • Continuous remote replication
		CLR	• Concurrent local and remote data • Combines CDP and CRR
	VMAX/VNX with a built-in RecoverPoint splitter	CDP/CRR/CLR	Both VMAX and VNX arrays have options with a built-in RecoverPoint splitter that functions as native continuous availability
	VMAX	SRDF/A SRDF/S SRDF/Metro	Continuous replication
	VPLEX	Metro	Active-Active between remote sites
	EMC MetroPoint™	CA and DR	3 or 4 sites configuration with CA and DR
Point-in-time rapid replication recovery	AppSync	Snapshot management interface for XtremIO, VMAX and VNX	• A simple, service-level agreement (SLA)-driven, self-service data protection, storage management for SQL Server • Also works with RecoverPoint on VNX • No agent necessary
	Replication Manager	Snapshot/Clone, SAN copy for VMAX and VNX	• A comprehensive data protection software • Must install agent on SQL Server
	VMAX TimeFinder	Mirror	General monitor and control operations for business continuance volumes (BCVs)
		Clone	Clone sessions generally consume the same size of production LUNs, but have no impact once created
		Snap	Snapshots consume less space than clones, but have more impact on production LUNs if the data changes frequently on the LUNs

Category	Tool/system	Features	Description
		VMAX3 SnapVX	SnapVX offers high-scale, deduplication-like, space efficient replicas, with unlimited cascaded capabilities, and no performance overhead.
	VNX	Clone	Clone sessions generally consume the same size of production LUNs, but have no impact once created
		Snap	Snapshots consume less space than clones, but have more impact on production LUNs if the data changes frequently on the LUNs
	XtremIO	Snapshot	Snapshots consume minimum space, and are read/writable, function like clones in a traditional hybrid array.
	XtremIO with RecoverPoint	CRR	Remote replication based on XtremIO snapshot

Each product has its own benefits and considerations. The decision depends on the service-level requirements of each use case.

EMC hardware-based snap and clone products have been integrated with Microsoft VDI and VSS technology for many years. Symmetrix TimeFinder and VNX SnapView (or advanced snap in the later version) enable local point-in-time snapshots and data clones for backup and recovery operations. These products enable simple, non-destructive backup operations with space-saving snapshots or full block-for-block clone copies of your databases and logs. With these products, backups and restores can occur in seconds.

EMC Replication Manager enables the management of EMC point-in-time replication technologies for SQL Server through a centralized management console. Replication Manager coordinates the entire data replication process—from discovery and configuration to the management of multiple, application-consistent, disk-based replicas. The databases can be automatically discovered with streamlined management for replication scheduling, recording, cataloging, and auto-expiration.

EMC strongly recommends a robust method that enables rapid SQL Server database backup and restore. EMC Replication Manager, Avamar, and EMC Networker® offer features for log truncation and the mounting of databases to alternative hosts.

Even if the native Microsoft SQL Server 2012 AAG is used, EMC strongly recommends an alternative, solid, point-in-time SQL Server data protection strategy to guard against logical corruption events.

General considerations

The most important requirements for implementing a multi-site disaster recovery solution usually in Service Level Agreements (SLAs) are:

- **Recovery time objective** (RTO): How long can the end user of the SQL Server tolerate an interruption in service.

- **Recovery point objective (RPO):** How much data loss can be tolerated.
- **Cost:** Cost of the solution to make that SLA feasible.

With a **synchronized replication solution**, data is only acknowledged when the remote site data is committed:

- **Advantage**: Zero data loss (RPO) at any time.
- **Disadvantage**: Synchronous replication over distances greater than 200 km may not be feasible.

An asynchronous replication solution will be committed before the remote site sends acknowledgement:

- **Advantage:** No limitation for replication distance.
- **Disadvantage**: It can have potential data loss.

The amount of data and volumes protected is another consideration of the multi-site protection design:

- All the data is replicated.
  - It can be set up on the remote site to automatically start when failover is needed, providing instant RTO.
  - More data transferring over the network might degrade production performance.
- Choose only user database and log files to replicate.
  - Less data transferring over the network means better production performance and less data loss.
  - Less data in the consistency group can potentially prolong the recovery procedure and time (longer RTO).

For the highest level of RTO and RPO, choose the synchronized solution, possibly with geographically dispersed clustering product, this will provide zero data loss solutions with extremely small RTO, with most processes automated (VMAX SRDF/CE). This needs invest on fast links between the sites.

For a higher level of RTO and RPO, VPLEX provides similar results with heterogeneous arrays at the remote site.

RecoverPoint, Replicator, and the rest of EMC multi-site replication technologies can all provide very good RTO and RPO with minimum user intervention to bring the remote site up when disaster recovery is needed.

EMC replication technologies considerations

The following sections include the best practices or considerations for EMC replication technology for continuous availability and disaster recovery options in Table 18.

EMC AppSync

AppSync orchestrates the creation and management of database replicas using underlying copy technologies that support various EMC storage platform and replication technologies.

AppSync provides simple, self-service application protection with tiered protection options and proven recoverability. AppSync protects an application by creating copies of application data.

For example, subscribing a SQL Server 2012 database to a service plan indicates to AppSync that you want to protect that database. When the service plan runs, one or more copies are created. The service plan can also mount the copy, validate it, and run user-created scripts. A service plan comprises multiple phases, including create copy, mount copy, validate copy, unmount copy, and phases that run optional user-provided scripts.

AppSync can generate reports that tell you whether your data is protected, recoverable, and compliant with SLAs. The reports included with AppSync work without modification. You can easily view alerts and reports at the top level of the AppSync dashboard. You can send alerts by email and export reports to comma-separated values (CSV) files.

AppSync components include the AppSync server software, host plug-in software, user interface, and REST interface:

- AppSync server software

  The AppSync server software resides on a supported Windows system. It controls the service plans and stores data about each copy it creates. The repository is stored in a SQL Server 2012 database on the AppSync server.

- Host plug-in

  AppSync installs lightweight plug-in software on the production and mount hosts. AppSync pushes the plug-in software from the AppSync server to the host when you add the host as a resource. In an environment that prevents the AppSync server from accessing a host, you can install the plug-in manually.

- AppSync user interface

  The AppSync console is web-based and supports Chrome, Internet Explorer, and Firefox browsers. Flash and Java Runtime Environment are required. The AppSync Support Matrix on EMC Online Support is the authoritative source of information on supported software and platforms.

- REST interface

  AppSync has a REST interface that allows application programmers to access information controlled by AppSync. The API is described in the *AppSync REST API Reference*, which is available on EMC Online Support.

VMAX3 TimeFinder SnapVX

VMAX3 TimeFinder SnapVX snapshots are consistent by default. Each source device can have up to 256 space-efficient snapshots that can be created or restored at any time. Snapshots can be further linked to up to 1024 target devices, maintaining incremental refresh relationships. The linked targets can remain space-efficient, or a background copy of all the data can take place, making it a full copy. In this way, SnapVX allows unlimited number of cascaded snapshots.

VNX Remote Protection Suite

VNX Remote Protection Suite delivers unified block and file replication, providing disaster recovery for both NAS and SAN environments. It delivers disaster recovery protection for any host and application without compromise—with immediate DVR-like rollback to a point in time. Capabilities include compression and deduplication for WAN bandwidth reduction, application-specific recovery point objectives, and replication options for one-to-many configurations.

- This suite for VNX series includes Replicator, MirrorView/A, MirrorView/S, and RecoverPoint/SE continuous remote replication (CRR).
- This suite for VNXe series includes replicator (iSCSI and NAS).

MirrorView replicates SQL Server database LUNs to remote locations for disaster recovery. MirrorView replication is transparent to the host. If the production host or the production storage system fails, the remote replication facilities fail over to the secondary mirror image.

MirrorView software offers two complementary mirroring products:

- MirrorView/S can synchronously mirror data images of production host LUNs to secondary storage at a remote site in real time. This offers zero data loss if there is a failure at the production site.
- MirrorView/A offers long-distance replication based on a periodic incremental update model. It periodically updates the remote copy of the data with all the changes that occurred on the local copy since the last update. This can result in data loss if there is a failure at the production site.

MirrorView works well in small- to medium-sized SQL Server environments. *EMC Business Continuity for Microsoft SQL Server 2008 Enabled by EMC CLARiiON* and *EMC MirrorView /A White Paper* on the EMC.com website provides more information about MirrorView.

SRDF/Metro

SRDF/Metro is available between VMAX3 family arrays. SRDF/Metro significantly changes the traditional behavior of SRDF synchronous mode with respect to the R2 device availability to better support host applications in high-availability environments.

With SRDF/Metro, the SRDF R2 device is also read/write accessible to the host and takes on the federated personality of the primary R1 device (geometry, device WWN, and so on). By providing this federated personality on the R2 device, both R1 and R2 devices may then appear as a single virtual device across the two SRDF paired arrays

for host presentation. With both the R1 and R2 devices being accessible, the host or hosts (in the case of a cluster) are allowed to read and write to both R1 and R2 devices with SRDF/Metro ensuring that each copy remains current, consistent, and addresses any write conflicts which may occur between the paired SRDF devices.

The VMware metro storage cluster (MSC) powered by EMC SRDF/Metro and vSphere HA can provide non-disruptive hardware maintenance and quick recover of data center failure. This provides the followings benefits of SQL Server protection:

- Enables SQL Server high availability across local and remote data centers
- Minimizes downtime during data center failure
- Enables SQL Server mobility across data centers without interruption

EMC MetroPoint

RecoverPoint 4.1 introduces a topology called MetroPoint, which protects a distributed VPLEX virtual volume at both VPLEX Metro clusters. A RecoverPoint cluster is attached to each VPLEX Metro site. One RecoverPoint cluster is the active source and the other RecoverPoint cluster is the standby source. The active source replicates data to the remote cluster. The standby source cluster does the same as the active source cluster, except that it does not replicate data to the remote cluster. The standby source cluster can instantly become the active cluster and continue data protection without disruption.

The MetroPoint topology provides protection in two-, three- and four-site SQL Server topologies.

Figure 17 is an example of a three-site SQL Server topology with MetroPoint. VPLEX Metro cluster 1 is deployed in New York, and VPLEX cluster 2 is in New Jersey. Above it, VMware Storage Metro Cluster is deployed across New York and New Jersey. Virtual machine HA and DRS is enabled to provide hypervisor-level high availability and automation resources allocation. SQL Server FCI works on two virtual machine instances across two sites (cluster across boxes or CAB) to provide application-level high availability.

A third site in Paris uses RecoverPoint cluster to connect to New York and New Jersey to provide disaster recovery function.

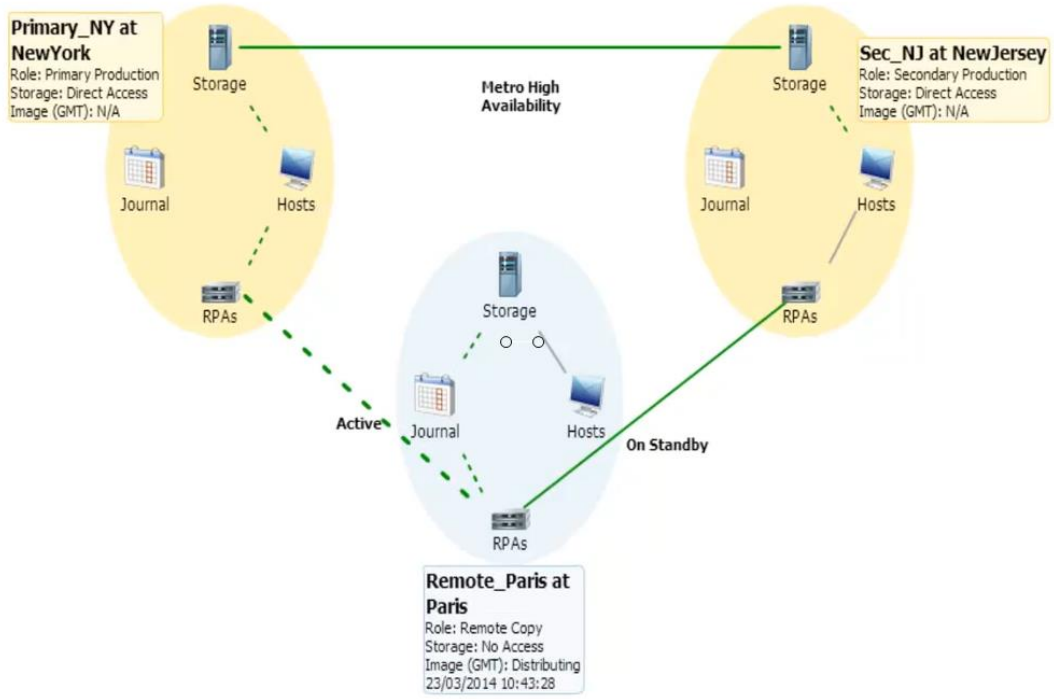

Figure 17. Three-site MetroPoint configuration for SQL Server FCI

The following three sections introduce the best practices for this configuration for the hypervisor, VPLEX, and RecoverPoint.

VMware vSphere considerations

EMC recommends that you use the following command to mark all the RDM devices used in WSFC as perennially reserved in each ESXi server to improve the rescan and ESXi reboot speed:

```
# esxcli storage core device setconfig -d naa.id --perennially-
reserved=true
```

SQL Server FCI virtual machines are configured in CAB mode, which requires that they are spread across different ESXi hosts. This requires configuring the virtual machine-virtual machine anti-affinity rule for SQL Server virtual machines, as shown in Figure 18.

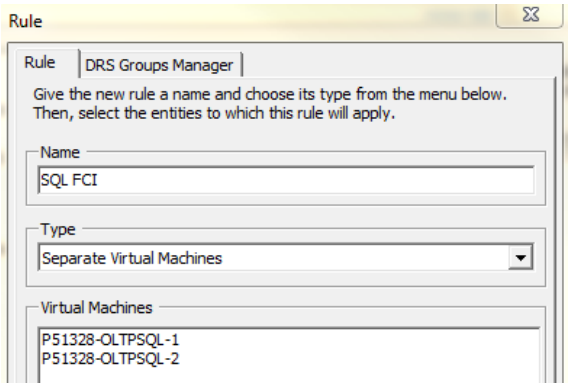

Figure 18. Anti-affinity rule set for SQL Server FCI

To ensure that the anti-affinity rules are strictly applied, set an advanced option for vSphere DRS. Setting the advanced option **ForceAffinePoweron** to 1 enables strict enforcement of the anti-affinity rules that you created.

EMC also recommends that you apply vSphere DRS groups and virtual machine-host anti-affinity rules to all the virtual machines across both VPLEX clusters.

VPLEX Metro considerations

VPLEX hardware connectivity best practices are based on a dual fabric SAN configuration, which is an industry best practice. In a dual fabric SAN, each unit must have connectivity to both Fibre Channel switches for redundancy.

When you configure the VPLEX cluster cabling and zoning, the best practice is to use a configuration that provides the best combination of simplicity and redundancy. In this solution, we use the following configuration:

- Implement dual fabric designs for fabric redundancy and HA to avoid a single point of failure. This provides data access even in a full fabric outage.
- Physically connect each VPLEX director to both fabrics for both host (front end) and storage (back end) connectivity. Hosts connect to both Director A and Director B from both fabrics and across engines. WAN ports of each director physically connect to both WAN fabrics for the supported HA level of connectivity, as required with the non-disruptive upgrade (NDU) pre-checks.
- The number of paths from each VPLEX director to a single storage LUN must not exceed eight or the performance will be affected.
- Ensure you have enough BB_Credits configured in FC switch WAN ports.

  For a more detailed explanation of VPLEX connectivity, refer to *EMC Technical Notes: Implementation and Planning Best Practices for EMC VPLEX.*

EMC suggests the following best practices in a VPLEX configuration:

- Create a valid distributed volume. The VPLEX virtual volume LUN number on both clusters must be the same for the devices that form the distributed volume.
- Set a large transfer size for migrations when data protection or migration performance is important. Set a smaller transfer size when front-end storage response time is more important

EMC RecoverPoint considerations

If an IP WAN is used between RecoverPoint clusters, EMC recommends:

- Put all related switch ports into a single VLAN.
- Set the MTU of the switch port and the RPA WAN port to 9,000 to accommodate jumbo frames.
- Set the size of the TCP/IP window properly according to the application I/O.

The best practice for RecoverPoint consistency group settings is to align with that of the VPLEX consistency group.

EMC recommends placing both the journal and the repository volumes on a VPLEX volume and creating a mirror so that a technology refresh on the underlying array is non-disruptive to the RecoverPoint replication. Because there will be many I/O writes to the journal volume, especially on the DR site, we recommend using RAID 1/0 and SAS disks to ensure good performance.

Virtualization considerations for disaster recovery

VMware vSphere

VMware Site Recovery Manager

VMware vCenter Site Recovery Manager (SRM) enables you to build, manage, and run reliable disaster recovery plans for your virtual environment. Site Recovery Manager takes full advantage of the encapsulation and isolation of virtual machines and enables simplified automation of disaster recovery.

Performance recommendations for SRM are as follows:

- Group SQL Server virtual machines under fewer protection groups. This practice enables faster recoveries, provided those virtual machines have no constraints that prevent them from being grouped under similar protection groups.
- Enable VMware DRS at the recovery site. This ensures optimal performance and recovery time because the recovered virtual machines are load balanced across the hosts by VMware DRS.
- If VMware DRS is not enabled, manually distribute placeholder SQL Server virtual machines evenly across the recovery hosts. This helps to distribute the load across hosts for all recovery virtual machines operations.
- Install VMware Tools in all protected SQL Server virtual machines to acquire heartbeats and network change notification.

Microsoft Hyper-V

SRDF/CE

SRDF with Cluster Enabler (SRDF/CE) facilitates the creation of remote, synchronous copies of mission-critical data that is also protected from loss by the presence of Microsoft Cluster Server software.

A minimum of one HBA is required for connectivity to the VMAX array. If PowerPath software is used, a minimum of two HBAs are required. For HBA driver recommendations, refer to the *EMC SRDF/Cluster Enabler Plug-in Product Guide*. For clustering reliability, EMC recommends that you have at least two paths per device.

EMC recommends allocating one additional gatekeeper on each host for each SRDF/CE group created for optimum performance. The gatekeepers must be associated on a per group basis and must be mirrored.

Chapter 6 Conclusion

This chapter presents the following topics:

Conclusion

Summary

This document highlights the key decision points in planning a Microsoft SQL Server deployment with EMC storage systems. Multiple configuration options are available to suit most environments. EMC storage and data management products are designed for flexible management of SQL Server environments to best meet your business needs.

Best practices for designing SQL Server storage are constantly evolving. With storage technology rapidly improving, traditional best practices may not apply to all configurations. This document presents the current best practices recommended by EMC for deploying SQL Server with the EMC XtremIO all-flash array, EMC Symmetrix VMAX storage series, EMC VNX family of unified storage, or EMC ScaleIO software-based storage platform. Following these guidelines can greatly assist you in achieving an efficient, high-performance, and highly available SQL Server environment that meets your requirements.

This paper presents the following concepts, principles, and formulas to help you:

- Understand the I/O and bandwidth characteristics of SQL Server.
- Apply best practices for SQL Server and the XtremIO, VMAX3/VMAX, VNX or ScaleIO storage series.
- Utilize a SQL Server storage building block.
- Calculate the storage I/O, capacity, and bandwidth requirements.
- Validate your overall storage design.
- Become familiar with the various data protection options for SQL Server.

Additional information

Consult your local EMC SQL Server expert for additional guidance on deploying Microsoft SQL Server with the EMC XtremIO, EMC Symmetrix VMAX series, EMC VNX family or EMC ScaleIO.

Chapter 7 References

This chapter presents the following topics:

Documentation

EMC documentation

These documents are available from www.EMC.com or EMC Online Support. Access to online support depends on your login credentials. If you do not have access to a document, contact your EMC representative.

- *EMC Virtual Infrastructure for Microsoft Applications enabled by Symmetrix VMAX and Microsoft Hyper-V White Paper*
- *SQL SERVER 2012 DATA WAREHOUSE: EMC VNX5500 HB (R32), VMware vSphere 5, Microsoft Windows 2012 White Paper*
- *EMC XtremIO Storage Array Host Configuration Guide*
- *EMC Converged Infrastructure for Microsoft Applications White Paper*
- *Managing Microsoft SQL Server Workloads by Service Levels on EMC VMAX3 Solution Guide*
- *EMC Technical Notes: Implementation and Planning Best Practices for EMC VPLEX*
- *EMC SRDF/Cluster Enabler Plug-in Product Guide*
- *Virtualizing SQL Server 2008 using EMC VNX Series and Microsoft SQL Server 2008 R2 Hyper-V White Paper*
- *Continuous Data Protection for Microsoft SQL Server Enabled by EMC RecoverPoint, EMC Replication Manager, and VMware White Paper*
- *EMC Replication Manager Administrators Guide*

VMware documentation

- *Using NUMA Systems with ESX/ESXi*
- *Configuration Maximums*
- *VMware ESX Scalable Storage Performance*
- *Troubleshooting Storage Performance in vSphere – Storage Queues*
- *Large-scale workloads with intensive I/O patterns might require queue depths significantly greater than Paravirtual SCSI default values (2053145)*.
- *Changing the queue depth for QLogic, Emulex and Brocade HBAs (1267)*
- *Setting the Maximum Outstanding Disk Requests for virtual machines (1268)*
- VMware Compatibility Guides

Other documentation

For additional information, see the documents or topics listed below.

Microsoft TechNet

- *SQL Server Books Online*
- *Storage Windows Server White Paper*
- *Hyper-V scalability in Windows Server 2012 and Windows Server 2012 R2*
- *Enable the Lock Pages in Memory Option (Windows)*

- *Hyper-V Virtual Fibre Channel Overview*
- *Database Engine Tuning Advisor*.
- *Using Files and Filegroups*

MSDN

- *Partitioned Tables and Indexes*
- *Restoring a Database to a Point Within a Backup*

Microsoft support

- *Tuning options for SQL Server when running in high performance workloads*
- *Recommendations to reduce allocation contention in SQL Server tempdb database.*
- *How to use the SQLIOSim utility to simulate SQL Server activity on a disk subsystem*
- *Storage: Windows Server 2012 White Paper*

Appendix A Optimizing the Storage I/O Queue for VMware vSphere Environments

This appendix presents the following topics:

Optimizing the storage I/O queue for VMware vSphere environments

This appendix describes the method for optimizing the storage I/O queue for the VMware vSphere environments.

Optimizing the World Queue

World Queue depth can be optimized by increasing the SCSI controller queue depth inside the Windows virtual machine. In this solution, we modified the number of pages that the controller uses for the request ring to 32, and increased the queue depth to 254.

To modify the number of pages and the queue depth:

1. On the Windows virtual machine, run the following script from the command line:

```
REG ADD
HKLM\SYSTEM\CurrentControlSet\services\pvscsi\Parameters\Dev
ice /v DriverParameter /t REG_SZ /d
"RequestRingPages=32,MaxQueueDepth=254"
```

Note: The changes apply to the VMware PVSCSI controller to service I/O patterns that require queue depth significantly greater than the default values (8 for request ring page, 64 for default queue depth). For more details, refer to the VMware KB topic *Large-scale workloads with intensive I/O patterns might require queue depths significantly greater than Paravirtual SCSI default values (2053145)*.

2. Reboot the virtual machine.
3. Confirm the creation of the registry entry for the parameters by navigating to the following path in the registry editor:

```
HKLM\
SYSTEM\CurrentControlSet\services\pvscsi\Parameters\Device
```

Optimizing the adapter queue

The HBA adapter queue configuration varies between different manufacturers. To support mission-critical OLTP and DSS workloads, the default values set by the manufacturer may not be adequate. In this solution, we adjusted the queue depth for the HBAs on the ESXi host on which the Windows virtual machines run. We also modified the default HBA throttle settings to improve I/O spike. For other parameters, accept the default settings.

To change the HBA queue depth in the ESXi 5.5 host:

1. Create an SSH session to connect to the ESXi host.
2. Verify the HBA module under-used:

```
# esxcli system module list | grep qln
```

3. Modify parameter **ql2xmaxqdepth** to 256:

```
# esxcli system module parameters set –p ql2xmaxqdepth=256 -
m qlnativefc
```

4. Reboot the ESXi host to make the changes take effect.

5. Run the following command to confirm your settings:

```
# esxcli system module parameters list -m driver
```

Note: These steps apply to QLogic native HBA drivers. For information about configuring the HBA queue depth for HBA drivers from other manufacturers, refer to the VMware KB topic *Changing the queue depth for QLogic, Emulex and Brocade HBAs (1267)*.

The following steps are an example of how to modify HBA I/O throttle settings. For the specific hosts, contact the server or HBA manufacturer for more details.

1. In the server management console, under **Server**, select **Inventory**, and then select **Cisco VIC adapters**.
2. Navigate to **vHBA Properties**.
3. Set **I/O Throttle Count** to "1024", as shown in Figure 19.

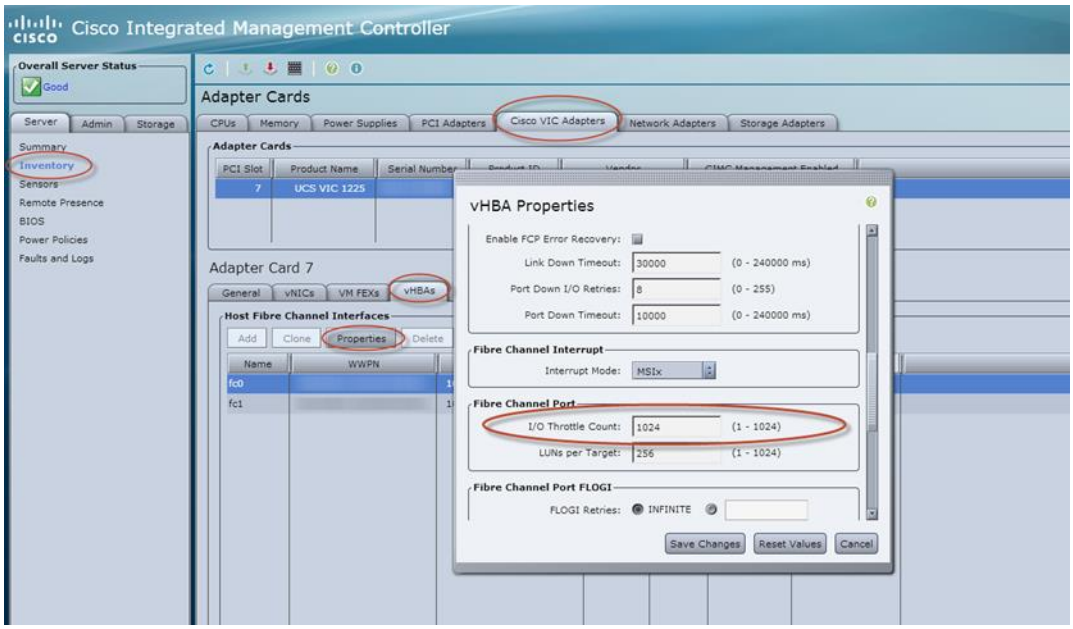

Figure 19. Modifying the HBA I/O throttle count

4. Click **Save Changes**.

Optimizing the device queue

For the device on the virtual machine, increase the value of maximum outstanding disk requests to improve disk intensive workloads.

To change the device queue depth:

1. Create an SSH session to the ESXi host.
2. Validate the current value for the World Queue parameter, by running the following script:

```
# esxcli storage core device list -d naa.id
```

Note: The value is displayed under **No. of outstanding I/Os with completing worlds**; default value was 32.

3. To modify the maximum outstanding disk requests to 256, edit the **Disk.SchedNumReqOutstanding** parameter by running the following script:

```
# esxcli storage core device set -d naa.id -O 256
```

Note: EMC recommends that you apply these steps for the devices for I/O intensive environments, especially for disks that service heavy IOPS or bandwidth. For more information, refer to the VMware KB topic *Setting the Maximum Outstanding Disk Requests for virtual machines (1268)*.