# Veritas Enterprise Vault™

## SQL Best Practices

14.1

VERITAS

# Veritas Enterprise Vault™: SQL Best Practices

Last updated: 2021-03-19.

## Legal Notice



Veritas Technologies LLC
2625 Augustine Drive
Santa Clara, CA 95054

https://www.veritas.com

# Contents

Chapter 1

# Introduction

## Purpose

Sizing and implementing Enterprise Vault requires careful planning to ensure that the product can perform and scale to expectations and ensure the underlying Enterprise Vault infrastructure is configured to support the required activity.

The Enterprise Vault SQL servers should be sized, tuned and maintained according to Microsoft best practice advice for SQL Server. See the TechNet article "SQL Server best practices". In addition, it is recommended to review all associated Microsoft tuning and update guidelines, including:

https://support.microsoft.com/en-us/kb/2964518

This guide discusses some of the SQL Server best practices from an Enterprise Vault perspective and should be used in conjunction with Microsoft advice.

The Enterprise Vault databases contain varied information including configuration information, item metadata, and reporting statistics. In many cases the data forms a key part of feature workflow and this can result in high workloads that will struggle to co-exist with any other database application. During Enterprise Vault implementation you must pay special attention to the SQL servers and their configuration.

This guide assumes that you are familiar with how to configure and administer Enterprise Vault, SQL Server and associated products. You can obtain more detailed installation and configuration information from the Enterprise Vault documentation. Veritas also publishes many white papers that explore specific Enterprise Vault details.

For advice on Discovery Accelerator sizing and tuning, see the *Best Practices for Implementation Guide*, which is available from the following page:

http://www.veritas.com/docs/100024378

# What's new in this guide

This guide has been updated from the previous version as the result of further performance investigations and feedback.

In most cases, the database considerations are equivalent to those of previous versions.

# Supporting documents

Use this guide in conjunction with the following documents:

- *Enterprise Vault Performance Guide*, which is available from the following page of the Veritas Support website:

  http://www.veritas.com/docs/100000918
- *Enterprise Vault Compatibility Charts*, which is available from the following page of the Veritas Support website:

  http://www.veritas.com/docs/000097605
- *Veritas Discovery Accelerator 14.1 Best Practices Guide*, which is available from the following page on the Veritas Support website:

  http://www.veritas.com/docs/100024378

  The guide discusses the different aspects that you need to consider during sizing and recommends best practices for implementation. Most of the advice in the guide applies to Compliance Accelerator as well.
- *Enterprise Vault Best Practice Guide - Implementing Enterprise Vault on VMware*, which is available at the following location:

  http://www.veritas.com/docs/10002381

Chapter 2

# Server requirements

The anticipated loads will help determine the best SQL Server edition and the number and size of SQL Servers that should be deployed. From an Enterprise Vault perspective, the key differences between the Standard and Enterprise or Datacenter editions of SQL Server are the scalability and high availability options.

Enterprise Vault is a feature-rich product and can therefore result in very diverse SQL Server load profiles between customer deployments. As Enterprise Vault scales, additional databases and the associated increase in load will require scaling the SQL Servers to meet the load.

It may be necessary to separate out specific Enterprise Vault databases such as the directory and audit databases to dedicated SQL Servers. We also recommend preparing dedicated SQL Servers with reporting services if FSA reporting is to be deployed.

There are many factors which may influence the deployment, but an initial server sizing guideline would be to provision the CPU cores and RAM described in the table below arranged in as many servers as desired, evenly distributing the Enterprise Vault databases if deployed as multiple SQL servers.

| | **SQL Server requirements** |
|---|---|
| Low load or no end-users (e.g. Journaling only) | Provision 4 cores and 8 GB RAM for every 5 Enterprise Vault servers |
| Up to 8 Enterprise Vault servers with normal load | Provision 4 cores and 8 GB RAM for every 3 Enterprise Vault servers |

| | SQL Server requirements |
|---|---|
| More than 8 Enterprise Vault servers with normal load – distributed databases | **Dedicated server for Directory, Audit, Monitoring & EV Reporting databases:**<br>Provision 8 cores and 16GB RAM for the first 8 Enterprise Vault servers plus an additional 4 cores and 8 GB RAM for every additional 12 servers.<br><br>**Dedicated SQL Server(s) to host other Enterprise Vault databases:**<br>**Vault Store & Sharing databases:**<br>Provision 8 cores and 16 GB RAM for every 8 Enterprise Vault storage services.<br>**ADSS database:**<br>Provision 2 cores and 8 GB RAM for every 8 Enterprise Vault indexing services.<br>**Note**: If any database is hosted on an isolated server the minimum should be 4 cores and 12GB RAM to accommodate operating system requirements. |
| FSA Reporting | **Dedicated SQL Server:**<br>Provision 2 cores and 4 GB RAM for every 8 file servers<br>**Note**: The minimum should be 4 cores and 12 GB RAM. |

For example, to support 16 Enterprise Vault servers and 64,000 users you might choose to implement:

- 1 × Enterprise edition server with 12 processor cores and 24 GB RAM for the Directory database.

Plus **one** of the following example combinations to host the Vault Store and Sharing databases:

- 4 × Standard edition servers each with 4 processor cores and 12 GB RAM.
- 2 × Enterprise edition servers each with 8 processor cores and 16 GB RAM.

- 1 × Datacenter edition server with 16 processor cores and 32 GB RAM (provided that the network and storage bandwidth can cater for the overall load).

Scaling Enterprise Vault to meet complex requirements, such as wide geographic distribution, may require deploying multiple Enterprise Vault installations (with independent directory and associated databases).

# Hardware considerations

We recommend the Enterprise Vault SQL servers run a single SQL Server instance only, with no other applications or services running on the servers. SQL Server needs to fully utilize the server resources, and another application may introduce contention issues that results in undesirable performance.

## CPU considerations

The power of a server is not necessarily determined by the CPU speed in terms of cycles per second. Factors such as the server architecture and number and type of processors and cores can provide a far greater benefit over increasing CPU speed.

Hyper-threading technology is claimed to typically provide up to 30% improvement in performance. These processors contain two architectural states on a single processor core, making each physical processor act as two logical processors. However, the two logical processors must share the execution resources of the processor core, so performance gains may not be attained and in some circumstances can even lead to degradation in performance.

Multi-core technology provides similar performance to a comparable multi-CPU server. These processors contain multiple complete processor cores, which act as complete physical processors. Each physical core has its own architectural state and its own execution resources, so the performance gains are reliable.

With the ever-increasing number of processor core combinations and high clock speeds, the traditional Front Side Bus architecture can start to become a bottleneck beyond eight processor cores. A popular and cost-effective method of scaling up the architecture is to use an architecture that supports non-uniform memory access (NUMA). Processors and memory are grouped into nodes that have high-speed local access. However, access to memory co-located with other processor nodes is slower. Therefore, the operating system (and potentially application software) needs to be NUMA-aware and

optimized to make the best use of processors, cores, and their associated resources. Windows server and SQL Server support NUMA. See the MSDN article "How SQL Server supports NUMA".

The recommended number of processor cores can be composed of either physical CPUs or similar combination of multi-core CPUs, but the sizing should not be based on hyper-threaded logical cores.

**Note**: Hyper-threading is not recommended to improve the database performance due to potential performance problems when the database places a load on the memory. See Microsoft Knowledge Base article 322385 for further information. If hyper-threading is to be used, particular attention should be paid to the MAXDOP setting as described in KB322385.

In most cases, the SQL Server instance should manage the CPU resources. Do not set the CPU affinity mask unless absolutely necessary, as this can significantly impact the performance. When you run multiple SQL Server instances, the most common reason for setting the CPU affinity mask is to prevent an instance being starved of resources (see "Multiple SQL Server instances" on page 38).

## Memory considerations

The recommended memory should be available at each SQL Server instance to ensure the data manipulation does not cause excessive paging to disk both in the Enterprise Vault databases and tempdb, which will quickly degrade the performance.

You must host the databases on 64-bit (x64 only) SQL Server editions. Using an x64-based platform provides more efficient memory utilization and brings many performance benefits.

Install the appropriate edition of Windows Server and SQL Server to support the capacity of memory you have installed. See the Enterprise Vault compatibility charts for supported versions of SQL Server.

See "Server settings" on page 45 for more information on tuning the memory.

## Network considerations

We recommend that the Enterprise Vault SQL servers and Enterprise Vault servers are connected via gigabit network technology. The SQL servers may require multiple network interface cards to support the anticipated loads.

The Enterprise Vault servers open a number of connections to each database, which vary according to deployment and loads. The number could be considerable, however.

The table below provides an indication of the typical number of connections to the databases.

| Database | Typical number of connections, depending on settings and loads |
|---|---|
| Audit | 1 per Enterprise Vault server |
| Directory | 15 per Enterprise Vault server |
| Fingerprint | 20 per Enterprise Vault storage service |
| FSA Reporting | 10 per Enterprise Vault FSA Reporting service |
| Monitoring | 1 per Enterprise Vault server |
| Vault Store | 120 per Vault Store |
| ADSS | 2 per Enterprise Vault indexing service and middle tier server |

To prevent network issues, we recommend that you disable the TCP Chimney Offload, TCP/IP Offload Engine (TOE), or TCP Segmentation Offload (TSO). For guidance on how to do this, see Veritas technical article http://www.veritas.com/docs/100023865.

# Storage considerations

It is vital to ensure the storage does not become a bottleneck. By following Microsoft SQL Server best practices, you can ensure that the SQL server is suitably sized. Avoid using network-based storage for the database files.

In most cases, you will need RAID-based storage to achieve your storage requirements. To maintain performance and reliability, consider hardware-based RAID rather than software-based RAID. To achieve redundancy on striped arrays while maintaining performance, consider the RAID scheme carefully.

RAID levels 5 and 6 are popular, cost-effective methods of achieving redundancy while maintaining striped disk read performance. However, writing incurs a cost of four to six physical operations per write. A poorly sized RAID-5 or 6 implementations can significantly reduce the performance of write-intensive activity. Correctly sizing a RAID-5 or 6 implementations to maintain write performance may become costlier than RAID-1+0, and therefore a RAID-1+0 scheme should be considered.

In the case of local or direct attached storage, use multiple controllers supporting multiple channels to distribute the load between the multiple storage locations and provide sufficient throughput. The controllers should also provide a battery-backed read and write cache to aid performance. A minimum of 512 MB controller cache is recommended for local or direct attached storage.

Before you use partitions on a storage area network (SAN), consider the I/O load together with any other applications that are already using the SAN to ensure that the performance can be maintained. Ideally, discuss the implementation with your SAN hardware vendor to ensure that you achieve optimum performance. Typically, you should create LUNs across as many suitable disks as possible, using entire disks rather than partial disks to prevent multiple I/O-intensive applications from using the same disks. When you configure HBAs on the host, ensure that the Queue Depth is set to an optimal value. This should be discussed with the storage vendor.

When you create a volume on a storage device (basic NTFS or virtual datastore), it is very important to align the volume with the device sector or stripe unit boundaries to prevent unnecessary disk operations, which can significantly impact performance. Most current operating systems and hypervisors create new volumes with an offset of 1 MB (2048 sectors).

---

**Note:** See the Microsoft TechNet article "SQL Server Best Practices" for more information on using the diskpart tool to create and align volumes.

---

This article also recommends that you format both log and data partitions with 64 KB allocation unit sizes.

In most cases you should only create a single volume on each physical disk array to avoid contention at the disks between the partitions.

Each database requires the disks to be arranged for two different purposes; the database data files and the transaction log files. The data files require good random access, and therefore a striped array of many disks should be used. The log files require good sequential write performance, so each log file should be placed on its own high-speed array with good transfer rates.

To achieve redundancy on the sequential write-intensive disks (log), use a RAID-1 or RAID-1+0 scheme with high speed, 15k rpm disks.

Arrange the SQL server storage to accommodate the different types of data, distributing the load as appropriate. The following types of storage might be considered for each data requirement:

Recommended partition types for SQL servers

| Partition | RAID type to achieve performance |
|---|---|
| System drive | As per Microsoft recommendations |
| **Log File location(s) for:**<br>Tempdb<br>Directory Database<br>Each Vault Store Database<br>Audit Database<br>FSA Reporting database log files<br>ADSS Database | RAID-1+0 array |
| **Data File location(s) for:**<br>Tempdb<br>Directory Database<br>Each Vault Store Database<br>Audit Database<br>FSA Reporting database data files<br>ADSS Database | RAID-1+0 array |
| Each fingerprint database log file | RAID-1+0 array |
| Each fingerprint database data files | 1 or more RAID-1+0 arrays to host the 32 filegroup data files |
| **Log File locations for:**<br>Monitoring Database<br>SQL Server Reporting<br>SQL Server Reporting TempDB | RAID-1, 5, 5+0 or 1+0 array |
| **Data File locations for:**<br>Monitoring Database<br>SQL Server Reporting<br>SQL Server Reporting TempDB | RAID-5, 5+0 or 1+0 array |

If multiple database files are located on one partition, it may require regular file defragmentation to maintain performance.

After you have created each Enterprise Vault database, consider altering the default data file size to a value that represents next year's probable activity.

This prevents file fragmentation and wasted I/O and waits while growing the files. See “Deployment and tuning” on page 33 for more information.

# Virtualized infrastructure

There are important aspects to consider when installing SQL Server in a virtualized infrastructure. Follow the recommendations of your hypervisor vendor and Microsoft when you size and configure the environment.

The primary objective is to ensure that the resource requirements described above are dedicated to the virtual machine to ensure minimum impact to the performance from intermediate layers or co-existing guests. The hypervisor should be type-1 (native) to ensure the minimum impact on hardware resource requirements.

Note the following general guidelines:

- In a typical virtualized infrastructure, local disks might be used for the hypervisor and SAN-based storage for the guest operating system images and data file locations. The guest operating system and data storage partitions should be independent dedicated locations, as described above.
- Disk partitions should be aligned with the device sector or stripe unit boundaries to prevent unnecessary disk operations that can significantly impact performance. By default, most current hypervisor versions align new partitions to 2048 sectors, which should be sufficient for most devices.

  The disk partitions to be used for the database log files should be created as recommended by the hypervisor vendor for sequential access.

  The disk partitions to be used for the database data files should be created as recommended by the hypervisor vendor for random access (most likely virtual hard disks).
- Virtual hard disks should be created as fixed size and not dynamic.
- Hyper-threading can be left enabled which may be used by the hyper-visor for some efficiencies; however, the hyper-threaded logical cores must not be counted in the resource allocation calculations. For example, a 32-core server with hyper-threading may show 64-logical cores. However, all calculations and resource reservations should still be based on 32-cores (do not allocate the additional 32-logical cores). This may appear to have spare capacity, but these additional cores do not provide the capacity of real cores and allocating these will be detrimental to performance.

- Avoid the use of virtual machine snapshots, which can impact performance. If snapshots are used for maintenance and upgrade purposes, the snapshots should be rolled-up before commissioning the server for production use.
- The memory requirements recommended above should be dedicated and prioritized to the virtual machine to prevent dynamic allocation or sharing.
- The number of processor cores as recommended above should be exclusively dedicated to the virtual machine, and the processor priority and bandwidth set to provide the virtual machine with full utilization.
- The server power management scheme and all associated hypervisor and operating system settings should be carefully considered to ensure any performance implications are understood or minimized. Using C-states is very likely to impact performance.

If you want to install the SQL Server instance on a virtualized machine, avoid installing multiple instances on the same virtual machine.

# Security product considerations

It may be necessary to use security products to protect company assets. However, without tuning, some products can be very invasive, and they can considerably affect performance. It is vital to ensure that any security product in use is tuned appropriately and that key disk locations are excluded from real-time scanning. See the following article for more information:

http://www.veritas.com/docs/100007613

Chapter 3

# Database capacity planning

The Enterprise Vault databases' storage capacity needs to be carefully considered. There are many factors that influence the size of the databases. In some cases, it is not practical to incorporate all of these factors into your calculations and achieve an accurate sizing, as the information is not readily available. The following sections describe how to make a high-level capacity estimate that provides some guidance, but in practice the actual sizes may vary.

## Initial and upgrade planning

The approach to capacity planning differs between initially sizing each database and upgrading from an existing version of Enterprise Vault. The calculations detailed in this chapter can be directly used for initially sizing each database for the version 14.1 schemas.

Enterprise Vault 12 introduces storage classification, which has negligible impact on vault store database capacity.

Enterprise Vault 12.1 introduces permanent NARA record exports, which adds a small overhead to vault store database capacity.

Enterprise Vault 12.2 introduces a new database to support Archive Discovery Search Services, and additional retention management records which add a small overhead to directory and vault store database capacity.

Enterprise Vault 12.3 introduces support for SMTP mailbox journaling, which has a slightly different vault store database storage requirement.

Enterprise Vault 12.4 introduces Auditing enhancements, which has a slightly different Directory and Auditing database storage requirements.

If you are upgrading to version 14.1, you must do the following:

- Recalculate the capacity requirements for each existing database using the existing volumes.

- Compare the capacity requirements with previous sizing calculations to determine if additional capacity is required.

The upgrade process may require significant additional storage capacity during the upgrade, most notably at the transaction logs and the tempdb database. The additional storage required during the upgrade depends on the existing database sizes, the upgrade path, the schema changes required, and database recovery models. This combination of factors makes it difficult to determine temporary storage requirements. However, the "Database upgrades" section on page 51 provides details on recommended upgrade procedures.

Before you upgrade the Enterprise Vault databases, review any existing database tuning and maintenance plans and take the appropriate backups. See "Database upgrades" on page 51 for more information.

# Directory database

The Enterprise Vault directory database stores details of the following:

- Enterprise Vault server configuration
- Enterprise Vault services configuration
- Enterprise Vault policies
- Content source server details and their configuration
- Target provisioning and synchronization configuration
- Client provisioning and IMAP mailbox/subscription configuration
- IMAP client activity statistics and historical log
- Archive and folder structure details
- Security identifiers and access control lists
- Index location and index volume configuration
- Storage sharing group, vault store and partition configuration
- Archive, PST and Index management tasks
- Archive legal hold details
- Enterprise Vault Extension content source and provider registrations
- Enterprise Vault Extension custom archive type registrations
- Retention Folder configuration
- Archive Discovery Search Services configuration
- Centralised Auditing configuration

The Directory database is used by all Enterprise Vault servers to co-ordinate all activities, and therefore can be under considerable load.

Use the following rule of thumb to size the Directory database:

Estimated capacity (MB) =
((24.7s)+(23.38u)+(92.3i)+((fa+fd)1.3)+(1.613fa)+(0.156fdp))/1000

Where:

| | |
|---|---|
| s | Total Enterprise Vault servers |
| u | Total mailboxes (Exchange, Domino, and Internet / SMTP Mailbox) |
| i | Total IMAP enabled mailboxes (Exchange and Internet) |
| f | Total file servers or SharePoint servers |
| a | Average archive points per file server or sites and sub-sites per SharePoint server |
| d | Average folders per file server or libraries/folders in all sites per SharePoint server |
| p | Average permissions per folder |

**Note**: This calculation provides a high-level estimate only that will vary by feature usage. It does not include growth from archive or folder changes and management tasks. In addition, it does not take into account the operating system, paging, and log file devices, and it does not include any additional capacity that may be required during product version upgrades.

The IO load will depend upon the total archiving and client loads across all Enterprise Vault servers and the SQL Server specification. However, the Directory data file on a 16 GB SQL Server under typical loads from 5 Enterprise Vault servers might average 20 IOPS and the log file average 100 IOPS.

The following typical example shows the estimated directory database size for a mailbox archiving environment with five Enterprise Vault servers managing a total of 20,000 mailboxes with 10% of users enabled for IMAP.

Estimated size (MB) = ((24.7s)+(23.38u)+(92.3i)+((fa+fd)1.3)+(1.613fa)+(0.156fdp))/1000

= (24.7*5)

+ (23.38*20000)

+ (92.3*2000)

+ ((0*0)+(0*0)*1.3)

+ (1.613*0*0)

+ (0.156*0*0*0)

/1000

=651.92 MB

Therefore, the estimated capacity of the Directory database needs to be 651.92 MB, plus 20% extra headroom. The total is 782.31 MB for the data file.

In the following large-scale example there are to be 20 Enterprise Vault servers which will be archiving a combination of Exchange mailboxes and File Servers. The Exchange archiving will be managing a total of 16,000 mailboxes (10% IMAP enabled). The file server archiving will be managing 80 files servers, each containing on average 350,000 folders that typically have 30 permissions on each folder. On average 850 archive points will be created on each file server.

Estimated size (MB) = ((24.7s)+(23.38u)+(92.3i)+((fa+fd)1.3)+(1.613fa)+(0.156fdp))/1000

= (24.7*20)

+ (23.38*16000)

+ (92.3*1600)

+ ((80*850)+(80*350000)*1.3)

+ (1.613*80*850)

+ (0.156*80*350000*30)

/1000

= 168,139.95 MB

Therefore, the estimated capacity of the Directory database needs to be 168 GB, plus 20% extra headroom. The total is 201.8 GB for the data file.

# Audit database

The Enterprise Vault audit database stores details of the following:

- Admin and synchronization configuration changes
- Archive activity (archive, view, delete, restore item) [detailed]
- Item changes to folder location or retention category
- Item recovery
- Item classification
- Domino, FSA and SharePoint activity, including item retrieval
- Migration activity (PST/NSF) [detailed]

- Advanced searches
- Move archive operations and tasks
- Indexing task operations
- Archive manual permission changes

All categories provide summary information and several categories can also provide more detailed information. Enabling per item auditing can cause the audit database to grow rapidly, and may also impact performance.

This database is created on the same SQL server as the Enterprise Vault directory database. However, the audit database can be moved to another server if required. If the database should be moved, the EVAudit ODBC system Data Source Name should be updated on each Enterprise Vault server.

See the following article for guidelines on how to do this:

http://www.veritas.com/docs/100016654

Use the following rule of thumb to size the Audit database. The base annual capacity should be included plus any additional capacity required for the selected audit levels:

| | |
|---|---|
| Base annual capacity (MB) | = (0.59r)+(2.359u) |
| +Admin audit annual capacity (MB) | = 88.1+(176.118u)+(176.118f) |
| +Archive item annual capacity (MB) | = (0.339mt)+(176.2a * 0.5c) |
| +View item annual capacity (MB) | = (88.06a)+(0.339mt)+(22.1u) |
| +Delete item annual capacity (MB) | = (88.1d) |
| +Tasks annual capacity (MB) | = (1.7m)+(0.678i) |
| +Migration annual capacity (MB) | = (88.1p) |
| +Manual permission capacity (MB) | = (88.1n) |
| /1000 | |

Where:

| | |
|---|---|
| u | Total users |
| r | Total archives |
| f | Average folders added/moved per year across all files servers or SharePoint |
| a | Average number of items archived per day |
| d | Average number of items expired/deleted per day |
| p | Average number of items imported via PST/NSF migration per day |

| | |
|---|---|
| i | Average number of indexes upgraded or rebuilt per year |
| m | Average number of archives moved per year |
| t | Average number of items per archive re-indexed or moved |
| c | Item classification disabled = 1<br>Item classification enabled = 2 |
| n | Average number of manual permission changes per day |

**Note**: This calculation provides a high-level estimate only. It does not take into account the operating system, paging, and log file devices. It also does not include any additional capacity that may be required during product version upgrades.

The IO load will depend upon the level of auditing, the total archiving and client loads across all Enterprise Vault servers and the SQL Server specification.

# Monitoring database

The Enterprise Vault monitoring database stores details of the following:

- Monitoring counter, collection and threshold configuration
- Gathered metric values and historic records
- The monitoring database will retain between 30 and 60 days of metrics sampled at a frequency of between 10 minutes to 1 hour from all monitored Enterprise Vault servers and journal archive sources.

Once the monitoring database has reached its configured maximum size it should then remain reasonably static.

Use the following rules of thumb to size the Monitoring database:

Estimate (MB) =
((46.88j)+(15.63e)+(9.38v)+(51.05s)+(r(1440/f)*((9.03j)+(1.88s)+(0.952e))))/1000

Where:

| | |
|---|---|
| s | Total Enterprise Vault servers |
| e | Total Exchange servers |
| j | Total journal mailboxes |
| v | Total Vault Stores |

| | |
|---|---|
| r | Retention period (30, 45 or 60 days – default is 30 days) |
| f | Sample frequency (10 to 60 minutes – default is 15 minutes) |

**Note**: This calculation provides a high-level estimate only. It does not take into account the operating system, paging, and log file devices. It also does not include any additional capacity that may be required during product version upgrades.

The IO load at the Monitoring database is likely to be low with regular peaks during statistic updates.

The following example describes an environment with five Enterprise Vault servers and a total of 10 vault stores, archiving 10 Exchange servers with 10 journal mailboxes at the default sample rate of every 15 minutes and retained for the default duration of 30 days.

```
Estimate (MB) =  ((46.88j)+(15.63e)+(9.38v)+(51.05s)+(r(1440/f)*((9.03j)+
                   (1.88s)+(0.952e))))/1000
                   = ( (46.88*10)
                   + (15.63*10)
                   + (9.38*10)
                   + (51.05*5)
                   + ( (30*(1440/15)
                    * ( (9.03*10)
                       + (1.88*5)
                       + (0.952*10) ) )
                    )/1000
                  =315.53 MB
```

Therefore, the estimated capacity of the Monitoring database needs to be 315.53 MB, plus 20% extra headroom. The total is 378.6 MB for the data file.

# Enterprise Vault Reporting database

Enterprise Vault reporting uses SQL Server Reporting Services to provide the reports. The reports, which can be customized, contain various details including:

- Enterprise Vault service and task status

- Volume of items archived per Enterprise Vault server
- Mailbox archiving status
- Archive quota usage per user
- Most frequently accessed archived items (requires audit database)
- Journal mailbox archiving status and trends (requires monitoring database)
- Vault store usage by archive or billing account

Some of the report content is generated from the Enterprise Vault monitoring or audit databases. See the Enterprise Vault documentation for more details of which reports require either the monitoring or audit databases.

The SQL Server Reporting Services and databases should be sized according to Microsoft recommendations.

# Vault store database

The vault store database stores details of the following:

- All archived items' metadata
- Fast Browse metadata stores
- Archive and content provider statistics
- Items queued for archiving and associated queue metadata
- Items requiring indexing, backup or synchronization of client applications
- Items requiring updates such as folder path or retention category
- Items requiring deletion or expiry
- Items currently in processing by the indexing services
- Items currently in processing by compliance sampling
- Item legal hold status details
- Records management tracking

The maximum number of items that may be stored in a single vault store database during its lifetime is 2,147,483,647. This includes items that have expired and are no longer in the database. Most customers are unlikely to reach this limit but a customer archiving a million items a day to a single Vault Store could reach this limit after 5 to 6 years of archiving irrespective of expiry policy. You may check how many items have been stored by running this SQL statement against the vault store database:

```
SELECT IDENT_CURRENT(N'Saveset') AS NextSavesetIdentity
```

In addition, a single archive cannot perform more than 2,147,483,647 data change operations such as archive, update and expire in a single Vault

Store. This is most likely to occur with journal archives, and the best method of managing this situation would be to roll over journal archives on a regular basis (perhaps by archived volume or date). You may check to find the most operations that have been performed for any archive in a Vault Store by running the following statement against the vault store database:

```
SELECT MAX(HighestIndexSeqNo) From ArchivePoint
```

In general, we recommend that a vault store does not contain more than 250 million items. This makes it easier to perform database maintenance activities such as rebuilding indexes and backing up databases.

The additional storage required for metadata stores introduced at 11.0 is included in the below “Archived item metadata (MB)” calculation. If upgrading from a version prior to 11.0 and enabling metadata stores on pre-existing data, calculate the additional capacity as follows. The additional database storage is only needed when Fast Browse or IMAP is deployed to users.

- Metadata stores (MB) = (0.73m + 0.44f + 0.63d)/1024
- Where:

| | |
|---|---|
| m | Total archived Exchange mailbox messages |
| d | Total archived Domino messages |
| f | Total archived file based items |

The Compliance Sampling in Storage, introduced at 11.0.1, temporarily stores item processing details during the process of capturing and transferring items to Compliance Accelerator. The typical operating overhead is incorporated into the base capacity below. However, to accommodate potential backlogs you should consider provisioning an additional 2 GB.

Use the following rules of thumb to size each vault store database:

| | |
|---|---|
| Base capacity (MB) | = ((m+j+d+n+t+f+s)*0.06)/1024)+8192 |
| Annual archived item metadata (MB) | + (1.25m+1.54d+2f+0.93t)/1024 |
| Annual journal item metadata (MB) | + (0.56j+0.74n+0.45s)/1024 |
| Annual export item metadata (MB) | + (0.098p)/1024 |
| Annual device metadata (MB) | + ((m+j+d+f+s)*v*0.05)/1024 |

Where:

| | |
|---|---|
| m | Total Exchange mailbox messages archived into vault store per year |
| j | Total Exchange journal messages archived into vault store per year |
| d | Total Domino mailbox messages archived into vault store per year |

| | | |
|---|---|---|
| n | Total Domino journal messages archived into vault store per year | |
| s | Total SMTP journal messages and/or Skype for Business conversations/conferences archived into vault store per year | |
| t | Total SMTP mailbox messages archived into vault store per year | |
| f | Total file based items archived into vault store per year | |
| p | Total permanent records exported using Export-EVNARAArchive | |
| v | Vault store storage device type, where: | |
| | Local/CIFS/SMB | = 0 |
| | Streamer | = 1 |
| | EMC Centera (without collections) | = 10 |

**Note**: This calculation provides a high-level estimate only based on typical data characteristics. It does not take into account the operating system, paging, and log file devices. It also does not include any additional capacity that may be required during product version upgrades.

The IO load will depend upon the archiving and client loads and the SQL Server specification. However, each Vault Store data file on a 16 GB SQL Server under typical loads might require 150 – 1,200 IOPS perhaps averaging 300 IOPS, whilst the log file IO load might average 100 IOPS.

# Fingerprint database

The vault store database stores details of the following:

- All sharable item content hash values
- All sharable item shared link references

Typically, all files or mail attachments larger than 20 KB will be identified as sharable and a document identity hash value added to the fingerprint database. For message-based archiving this normally represents around 20% of the messages. File based archiving of office type documents are likely to see a similar proportion, but non-office files may vary considerably depending upon the data source.

Use the following rule of thumb to size the Fingerprint database. It assumes a message sharing ratio of 2 and file sharing ratio of 1.2.

Capacity per year (MB) = (((pm/2)+(f/1.2))*0.5)+((m+f)*v*0.05) /1024

Where:

| | |
|---|---|
| m | Total messages archived per year in all vault stores in sharing group |
| p | Percentage of messages with attachments eligible for sharing (assume 0.2 if unknown) |
| f | Total files archived per year in all vault stores in sharing group |
| v | Vault store storage device type, where:<br>Local/CIFS/SMB/Centera = 0<br>Streamer = 1 |

**Note**: This calculation provides a high-level estimate only. It does not take into account the operating system, paging, and log file devices. It also does not include any additional capacity that may be required during product version upgrades.

The IO load will depend upon the total archiving and client loads across all Vault Stores in the sharing group and the SQL Server specification. However, the Fingerprint data files on a 16 GB SQL Server under typical loads from 5 Enterprise Vault servers might average 80 IOPS and the log file average 120 IOPS.

As the Enterprise Vault requirements scale it may be necessary to distribute the I/O load of the fingerprint database. The Fingerprint database partitions its tables across 32 file groups and these can be distributed across multiple disk partitions during creation using the “New Vault Store Group” wizard. If changes to the locations are required after deployment, the filegroups would need to be moved manually within SQL Server.

# FSA Reporting database

The FSA Reporting database stores details of the following:

- Server and drive statistics
- Folder and file statistics
- User statistics

The FSA Reporting databases enable the data analysis reports to be generated and are updated by the Enterprise Vault File Collector service. The SQL Server Reporting Services and databases should be sized according to Microsoft recommendations.

If required, at least one FSA Reporting database should be installed per directory database. Multiple FSA Reporting databases can be created on separate SQL Servers to scale-out the anticipated loads. Multiple FSA

Reporting databases can also be used to segregate data, for example by geographical region.

Ensure that the file servers are evenly distributed between the FSA Reporting databases. The amount of data that the File Collector server must upload from a scan of a file server increases with the number of archive points and volumes that require scanning.

A data upload bottleneck can result if you assign many file servers that have the same FSA Reporting scan schedule to the same FSA Reporting database. If possible, distribute file servers with the same scan schedule to different FSA Reporting databases. Alternatively stagger the FSA Reporting scan schedules for the file servers that are assigned to the same FSA Reporting database.

---

**Note**: The FSA Reporting databases require special maintenance attention to ensure they remain within acceptable sizes.

---

FSA Reporting creates a SQL Agent job to move data from current tables to historical tables, by default at 9PM each day, but if the SQL Agent or purge job is disabled the current tables could grow unconstrained.

The historical tables will also require regular manual trimming to prevent unlimited growth. See the Enterprise Vault documentation for more information on using the FSAReporting_TrimData.bat utility to trim the historical tables. It is recommended to keep all historical tables to within 30 days data.

Use the following rules of thumb to size the FSA Reporting database:

Estimated (MB) =
((4.248f)+(0.339fv)+(2.218fr)+(0.536tr)+(0.61fvur)+(0.11fvr))/1000

Where:

| | |
|---|---|
| s | Total number of file servers |
| u | Average number of users per file server |
| v | Average volumes per file server |
| t | Total number of unique file types |
| r | Historical retention period (days – recommended 30) |

**Note**: This calculation provides a high-level estimate only. It does not take into account the operating system, paging, and log file devices. It also does not include any additional capacity that may be required during product version upgrades.

The IO load will depend upon the file collector loads and the SQL Server specification. However, each Vault Store data file on an 8 GB SQL Server under typical loads might require 150 – 1,200 IOPS perhaps averaging 300 IOPS, whilst the log file IO load might average 100 IOPS.

# ADSS database

The Archive Discovery Search Service database stores details of the following:

- Enterprise Vault configuration details
- Submitted search details and status
- Index volume search requests in progress
- Search result URLs awaiting retrieval

The ADSS database provides co-ordination and management between the middle-tier components and the individual ADSS search brokers.

Use the following rules of thumb to size the ADSS database:

| | |
|---|---|
| Base capacity (MB) | = (0.38u+33.84v+27.08a0.24s+2.59sv)/1000 |
| Annual search metadata (MB) | = (53.69s)/1000 |

Where:

| | |
|---|---|
| v | Total number of indexing servers |
| u | Total number of archives |
| s | Average searches per month |
| a | Average number of archives per search |

**Note**: This calculation provides a high-level estimate only. It does not take into account the operating system, paging, and log file devices. It also does not include any additional capacity that may be required during product version upgrades.

The IO load will depend upon the ADSS loads and the SQL Server specification. However, the ADSS data file on a 16 GB SQL Server under typical loads might average 100 IOPS, whilst the log file IO load might average 20 IOPS.

Chapter 4

# Deployment and tuning

The SQL server and databases require additional tuning to ensure the best performance is achieved.

## Deployment

### SQL permissions

The Enterprise Vault administration features enable Enterprise Vault administrators to create various databases. This is dependent on the Vault Service account having the SQL server role of database creator (dbcreator).

The Enterprise Vault databases are created with the Vault Service account as the DBO which provides sufficient permissions for normal operation.

Enterprise Vault, Compliance Accelerator, and Discovery Accelerator provide database roles with which you can control SQL database security.

You can use these roles to grant the Vault Service account only the permissions needed for normal daily operations, and additional permissions when they are required. See the following article for more information:

http://www.veritas.com/docs/100038151

### Deploying databases and the impact of model database

SQL Server creates new databases based upon the model database, so any changes to the model database will also be present in Enterprise Vault databases that are subsequently created. In addition, some of the default model database values may not be appropriate for some of the Enterprise Vault databases, for example, options such as file autogrowth values.

Therefore, after creating any Enterprise Vault databases, some of the options will need to be checked. The following settings and options can be examined and changed by either opening the database properties in SQL

Management Studio or through the use of database views, sp_dboption and ALTER DATABASE (see SQL Server Books Online).

- The database transaction log file autogrowth should be set around 200 MB – 1 GB for each Enterprise Vault database. The database data file autogrowth value should be set according to the recommended values in the table below. This can be viewed using SQL Management Studio database properties or the following SQL statement could be used to gather the sizes (in 8 KB pages):

```
SELECT name, type_desc, physical_name, size, growth,
max_size, is_percent_growth from sys.database_files
```

Recommended auto growth values for databases

| Database | Data file autogrowth value |
|---|---|
| Directory | Approximately 1 to 2 weeks' growth as per directory sizing for synchronization. For example, 200 MB growth, unlimited growth. |
| Vault Store | Approximately 1 to 2 weeks' growth as per sizing for daily archiving. For example, 500 MB growth, unlimited growth. |
| Fingerprint | 50 MB growth per filegroup, unlimited growth |
| Monitoring | 200 MB growth, unlimited growth |
| Audit | 300 MB growth, unlimited growth |
| FSA Reporting | 200 MB growth, unlimited growth |
| ADSS | 200 MB growth, unlimited growth |

- The database recovery model should be set according to your backup or high availability strategy. The recommended default is FULL. The current value can be viewed using SQL Management Studio database properties or using the following SQL statement:

```
SELECT name,(CASE recovery_model WHEN 1 THEN 'FULL' WHEN 2
THEN 'BULK_LOGGED' WHEN 3 THEN 'SIMPLE' END) from
sys.databases
```

- The database options should be checked with the expected default values. The current options can be viewed using SQL Management Studio database properties dialog or the following SQL statements will show the options set:

  ```
  EXEC sp_dboption <database name>
  ```

  Or, for SQL Server 2014 SP3 onwards, use:

  ```
  SELECT * FROM sys.databases where name = '<database name>'
  ```

  Only the AUTO_CREATE_STATISTICS and AUTO_UPDATE_STATISTICS options should be set. Any other options set should be returned to their default using either the SQL Server Management Studio database properties dialog, or using ALTER DATABASE (see SQL Books Online).

When Enterprise Vault creates each database, it will be created with the following default size. It is recommended to increase the data device size to either the first year's growth or at least the expected initial growth. This prevents file fragmentation, wasted I/O and waits growing the files during the initial, potentially high loads, archiving any backlog.

| Partition | Transaction Log | Data device |
|---|---|---|
| Directory | 25 MB | 10 MB |
| Vault Store | 80 MB | 100 MB |
| Fingerprint | 80 MB | Primary filegroup 100 MB<br>Non-primary filegroups 32 × 1 MB |
| Monitoring | 80 MB | 100 MB |
| Audit | 80 MB | 100 MB |
| FSA Reporting | 80 MB | 100 MB |
| ADSS | 1024 MB | 1024 MB |

## Co-locating Enterprise Vault databases

The Enterprise Vault databases can be co-located on the same SQL server, provided the appropriate resources are available; however, hosting many very active databases on a single SQL server can become detrimental to performance.

When co-locating multiple databases on a single SQL Server, it is important to ensure that the expected I/O loads can be accommodated by the attached

storage. This may require distributing the database files between multiple storage arrays to isolate the loads.

You must size the SQL server appropriately to host the required load. The cost of scaling up a single server can become less effective than using multiple SQL servers.

The Enterprise Vault databases should be distributed between the number of SQL servers identified in the sizing exercise. The sizing exercise may have identified the need for a dedicated server for the Directory database and FSA Reporting databases. If per-item auditing is required and there are more than 5 Enterprise Vault servers, it would be worth considering hosting the audit database on a dedicated SQL server.

It is recommended not to co-locate non-Enterprise Vault databases on the same SQL server.

# Database tuning

## Server updates

It is very important to follow Microsoft recommendations on updating the installed version of SQL Server to ensure reliability and performance is maintained. For SQL Server 2014 SP3, see Microsoft article https://support.microsoft.com/en-us/kb/2964518.

## Server settings

In most cases the majority of server settings should be left at their default values. In some cases, specific tuning may be required. Particular care should be taken when changing these settings to ensure changes do not severely affect performance.

### Memory and processor

The memory maximum value should be tuned to ensure sufficient physical memory is available for the operating system and any co-existing instances, reporting services, or other services. A simple rule of thumb for a single instance would be to set the maximum memory to 75% of the available RAM.

Processor and I/O affinity should only be tuned according to Microsoft advice when absolutely necessary, for example when using multiple SQL Server instances.

### Advanced settings

It is very important that the Maximum Degree of Parallelism should be tuned according to the recommendations in the following Microsoft article, taking into account the number of cores, hyperthreading and the number of cores per NUMA node ensuring any virtualization correctly presents NUMA to the operating system:

http://support.microsoft.com/kb/2806535

See also the following article for more information on hyper-threading:

http://support.microsoft.com/kb/322385

## The tempdb database

The tempdb database is a shared resource that is used to store the following:

- User objects (user defined objects, temporary tables and indexes, table variables, tables returned in functions, and system tables).
- Internal objects (work tables for cursors, spool operations, LOB storage, hash join storage, hash aggregate storage, and intermediate sort results).
- Version stores (row versions from updates).
- The tempdb database is a temporary database containing transient data that is recreated on each restart of SQL Server, so it will not need to be recovered.

The tempdb database pages move quickly between memory and disk, which becomes a bottleneck if the disks are inappropriate and the configuration is not tuned. You can take the following steps to avoid problems:

- To ensure that tempdb does not become a bottleneck, create one data file per CPU core, taking into account CPU core affinity mask settings. Make each file the same size as below.
- To ensure that tempdb has sufficient capacity to execute expected loads and to prevent tempdb file growth from causing unnecessary I/O or fragmentation, create each data file with the following size and growth:

| Feature | Recommended |
|---|---|
| Initial size | 200MB x *number of databases hosted on the server* |
| Growth (fixed) | 200MB + (20MB x *number of databases hosted on the server)* |

- Move the tempdb data files to a dedicated striped disk array (RAID-1 or 10) and the log file to a dedicated high-speed drive with high transfer rates (RAID-1 or RAID-10). You can use the following SQL statement to move the file location (of the default data file), which will take effect on the next SQL Server restart:

```
USE master;

GO

ALTER DATABASE tempdb

MODIFY FILE (NAME = tempdev, FILENAME =
'E:\SQLData\tempdb.mdf');

GO

ALTER DATABASE tempdb

MODIFY FILE (NAME = templog, FILENAME =
'F:\SQLData\templog.ldf');

GO
```

  **Warning**: The tempdb database is recreated each time SQL Server is started. If the new filenames are invalid, SQL Server will fail to start. When this happens, you must start SQL Server from the console with the –f flag to enter single user mode. Then you can repeat the above procedure with a valid location.

- Do not replicate the tempdb storage device or configure SQL Server replication on the tempdb database. The tempdb database is a temporary database containing transient data that is recreated on each restart of SQL Server, so it will not need to be recovered (use the SIMPLE recovery model). Any form of replication can significantly impact the performance.

# Multiple SQL Server instances

Running multiple SQL Server instances on the same server is not recommended because of the load profile of Enterprise Vault. However, if this is unavoidable, you should tune the instances appropriately to reduce any impact between them and ensure that they are not starved of resources.

The SQL Server instance that hosts the Enterprise Vault databases should have the minimum memory option configured to ensure that the memory recommended in the “Memory considerations” section on page 12 is dedicated to it.

To ensure that the SQL Server instance is not starved of CPU resource, you may need to consider balancing the available CPUs between the installed SQL Server instances. There are several ways to do this, including the following:

- Using the Windows resource manager to allocate resources to each instance.
- Setting the CPU affinity mask of each SQL Server instance to bind each SQL Server scheduler to specific CPUs.

You can also limit a SQL Server instance to a subset of CPUs without binding the SQL Server scheduler to specific CPUs by using trace flag 8002 with CPU affinity. This should counteract any performance problems that are associated with using CPU affinity. For more information, see the Microsoft Press book on SQL Server tuning.

The hardware architecture needs to be considered to ensure CPUs are not inefficiently allocated to the SQL Server instances, which can be significantly detrimental to performance.

In the case of NUMA architectures, the processors, cores, and memory are arranged in nodes that should be carefully allocated to the SQL Server instances to avoid creating a memory or I/O bottleneck. It may be worth considering using Soft-NUMA to divide the resources with both non-NUMA and NUMA-based hardware. See the Microsoft Press books for more information.

## Reporting services

Reporting services may be required for Enterprise Vault reporting or FSA Reporting. If the reporting services are not located on a dedicated SQL Server, it may be necessary to tune the memory requirements for both the SQL Server engine and Reporting services to ensure the available resources are appropriately shared.

For more information on configuring reporting services memory limits, see the MSDN article “Configuring Available Memory for Reporting Services”.

# Advanced tuning

There are circumstances in which the SQL Server may benefit from additional tuning; attempting more advanced tuning may or may not be beneficial.

## Index tuning

Enterprise Vault databases are supplied with indexes that are optimized to provide a balance between storage requirements and performance for typical data and feature utilization.

Making changes to existing indexes or adding new indexes is not currently supported. It is vital that existing clustered indexes are never changed, and clustered indexes are not added to heaps.

An experienced database administrator may identify indexes that are beneficial to specific customer data and loads, and in this situation the recommended index should be discussed with Veritas Support. When considering indexes, take care to ensure that data manipulation performance is not impacted, and the storage requirements are not excessive. Index read/write ratios will vary considerably between each table and according to specific customer feature usage and loads throughout the week.

If Veritas Support does recommend any customer-specific changes, you may need to remove them before Enterprise Vault upgrades. You may also need to reconsider them before you apply them to the new version, which may have changed schema and performance characteristics.

## Distributing I/O with multiple database files

Adding multiple database files to Enterprise Vault databases should not be necessary and the recommended approach is to ensure the Enterprise Vault database files are placed on an adequately sized storage array.

## Moving tables or indexes into file groups

Splitting tables and indexes into separate file groups can add considerable complexity to the database management. Any errors during implementation can lead to data loss, damage to the schema, and problems during product upgrades. Therefore, making changes to the existing schema is not recommended.

However, the Fingerprint database does partition tables across 32 file groups and these can be distributed across multiple disk partitions using the “New Vault Store Group” wizard. If changes to the locations are required after deployment, the filegroups would need to be moved manually within SQL Server.

Chapter 5

# High Availability

Enterprise Vault 14.1 supports SQL Server 2014 SP3 and onwards Always On Availability Groups and Failover Cluster Instances, which make use of Windows Server Failover Clustering. Using these technologies, Enterprise Vault automatically handles failover situations.

This chapter describes the recommended practices for deploying Enterprise Vault databases for High Availability using Always On technologies. The chapter does not consider any previous SQL Server HA methods, such as database mirroring or log shipping or any other server clustering technologies such as Microsoft Cluster Services.

The chapter assumes that you are familiar with installing and configuring Failover Cluster Instances or Availability Groups, and associated Microsoft recommended best practices.

## Introduction

Failover Cluster Instances provide SQL Server instance-level continuity over multiple servers through shared database access. Enterprise Vault databases can be added to Failover Cluster Instances to provide highly available SQL Server and associated database access.

In most cases Failover Cluster Instances provide the most efficient solution for providing high availability.

Availability Groups provide database-level continuity over multiple SQL Servers through database replication. Enterprise Vault databases can be added to Availability Groups to provide highly available database access, and various replica scenarios.

However, Availability Groups introduce complexity, increased storage requirements and many additional configuration considerations to ensure the database reliability and performance is maintained.

# Preparation

When preparing the SQL Server environment for Enterprise Vault databases, it is vital to ensure that every SQL Server instance participating in the infrastructure is consistent. In particular, ensure that the following aspects are identical between each instance:

- Each SQL Server version, service pack and patch level is the same. It is important to take into consideration Microsoft updates to ensure the FCI or AG is reliable. There are a number of critical known issues in Availability Groups with several Microsoft Cumulative Updates. For more information, see the following Microsoft article:

  https://support.microsoft.com/en-us/kb/3033492
- Each instance is configured with the same collation.
- Each instance is configured with the same logins and permissions for Enterprise Vault.
- All the availability replicas have the same service master key.
- Each instance is tuned according to Microsoft recommendations and the “Deployment and tuning” chapter on page 33.
- If using a non-default SQL Server port (not recommended), each instance uses the same port.
- Each SQL Server instance uses the same drive letters, particularly for locations such as database files and backup locations.
- Each SQL Server participating in the HA infrastructure has the same specification.
- Do not employ Multi-Subnet Failover clustering; this is currently not supported.

## Availability Groups

When configuring Availability Groups, there are a number of considerations in respect of Enterprise Vault databases and their maintenance, detailed in the sections below.

Enterprise Vault 14.1 does not support Availability Group read-only routing and does not provide additional scalability options utilizing multiple secondary replicas.

However, read-only replicas can be created for purposes such as offline reporting queries and other HA/DR purposes. If this is required, in most cases it is recommended to create asynchronous read-only replicas to minimize the impact to EV database performance.

The server sizing will need to accommodate the storage requirements for each database replica.

As with any replication technology, backups are still required to prevent data loss.

It is recommended to use a static IP address for the AG listener.

# Deployment

Creating new Enterprise Vault 14.1 databases in a Failover Cluster Instance is simply a matter of creating the new databases using the relevant virtual IP address, ensuring the database files are created on the appropriate storage device.

To create new Enterprise Vault 14.1 databases in an Availability Group, first create them using the relevant AG listener address. Then manually add the databases to the Availability Group using SQL Server management tools or the PowerShell cmdlet Add-SqlAvailabilityDatabase.

Migrating existing Enterprise Vault databases to a Failover Cluster Instance or Availability Group will require planning to ensure an appropriate migration sequence is developed and Enterprise Vault database connection configuration settings changed using the new PowerShell cmdlets. See the Enterprise Vault *Administrator's Guide* for more information.

Once databases are created or migrated to Availability Group SQL servers, it is crucial to add them to the Availability Group as soon as possible to ensure that the databases are available on failover.

## Availability Group considerations

To ensure the performance of the databases is sufficient, the synchronization mode and locality of each replica must be carefully considered.

It is recommended to create primary and synchronous replica databases with automatic fail-over locally to the Enterprise Vault infrastructure to ensure performance is not degraded.

Enterprise Vault monitoring will report on the most recent database backup from all replicas in an Availability Group. The reporting does not distinguish between whether the replica is synchronous or asynchronous.

Enterprise Vault should always be configured to connect to the Availability Group listener to prevent downtime on failover.

## SQL permissions

The Enterprise Vault login on each SQL Server hosting the Availability Group will need an additional server level permission “View Any Definition”.

## Directory and audit database

By default, the audit database uses the same connection details as the directory database and assumes that the audit database will be co-located with the directory database.

Therefore, when the audit database is created, it will be created in the same location as the directory but will still need to be manually added to an Availability Group. It is recommended to add the audit database to the same Availability Group as the directory database to ensure that the audit database is available on failover.

## Synchronous replica considerations

Each synchronous secondary replica must have committed each transaction log entry before the primary replica can commit the transaction. This can introduce considerable performance impacts, particularly if replicas are located over slow network connections or on lower specification hardware than the primary.

This could also have significant performance impacts in case of automatic fail-over.

To minimize the performance impact of synchronous replicas and ensure that automatic failover provides consistent performance, consider the following:

- Ensure the primary and synchronous replicas are co-located on a high-speed network. Do not create synchronous replicas over slow connections.
- Ensure each SQL Server is the same specification.
- Keep the number of synchronous replicas to the absolute minimum required.
- As with any replication technology, the replica will also contain any errors introduced in the primary, so a backup strategy is essential to prevent data-loss.

## Asynchronous replica considerations

An asynchronous replica will not prevent the primary replica from committing transactions. However, the following should be considered:

- Each additional replica will generally add an overhead, which can increase network and server loads. Therefore, keep the number of replicas to the minimum.
- The asynchronous replica is not guaranteed to be up to date, so should not be relied upon for high availability, which could lead to data-loss. However, this might form part of a disaster recovery strategy, ensuring you take into consideration errors might be replicated.
- As with any replication technology, the replica will also contain any errors introduced in the primary, so a backup strategy is essential to prevent data-loss.

## SQL Agent and maintenance jobs

SQL Agent jobs are not part of Availability Groups and are not replicated, automatically synchronized or failed over.

Therefore, you must ensure that any maintenance agent jobs created are manually copied to each SQL Server that is part of the availability group and set up to be appropriately synchronized.

Each Agent job must first identify whether the local replica is the current primary before performing any maintenance or other tasks. This can be achieved using the SQL function `sys.fn_hadr_is_primary_replica` as a condition in the maintenance job.

Any Agent jobs created by Enterprise Vault will be automatically created on each replica server and synchronized. However, if an Enterprise Vault Agent job scheduling is manually modified on the primary using SQL Management tools, it will not be replicated by SQL Server to the other agents and therefore must be manually adjusted on each replica.

## Backup jobs

An appropriate backup strategy is essential to prevent data loss, in particular where errors such as dropping a table would be replicated.

Typically, an Availability Group should be configured to prefer backups on a secondary replica.

SQL Agent jobs are not part of availability groups and are not replicated, automatically synchronized or failed over. Therefore, you must ensure any Agent jobs are manually copied to each SQL Server that is part of the availability group and set up to be appropriately synchronized.

Backup jobs created using the Maintenance Plan Wizard natively use the `sys.fn_hadr_is_preferred_backup_replica` function to determine which server to perform backup on. For other backup jobs Microsoft recommends

that you use this function as a condition in your backup jobs, so they execute only on the preferred replica.

In addition, the backup job will need to take account of any local server differences such as differing backup locations. This can be simplified by ensuring every SQL server has the same resource configuration.

## Deleting or moving databases

Before Enterprise Vault databases can be deleted or moved they must be removed from the Availability Group.

---

**Note**: Reorganizing replicas within an availability group does not require removing the database from the group and can be achieved as per Microsoft best practice advice.

---

Chapter 6

# Maintenance

Implementing an Enterprise Vault database maintenance plan is essential to preventing data loss, protecting data integrity and preserving the databases performance.

## Maintenance plan

It is recommended that regular backups of all Enterprise Vault databases are performed, including when using Availability Groups or Failover Cluster Instances.

When performing backups, the Enterprise Vault services should be placed into read-only/backup-mode, and all other associated Enterprise Vault data backed up to ensure consistency across the backups.

When using Availability Groups, backups should ideally be taken from secondary synchronous replicas.

After the backup has completed, the database log files could be shrunk.

As the databases are used, crucial tables and their indexes fill and can become fragmented. This reduces overall performance and can lead to additional I/O. The solution to fragmented indexes is to reorganize or rebuild very active indexes regularly. Index maintenance should be executed whilst Enterprise Vault servers are in read-only/backup-mode.

In addition, as some tables grow very large, the frequency of statistics update may become less than desirable. This can result in out-of-date statistics and potentially cause inefficient query plans. Therefore, it is advisable to update the statistics on a regular basis.

Create a regular maintenance plan that includes the following tasks:

- Performing the Enterprise Vault database backup
- Reorganizing indexes
- Updating statistics

See the following article on the Veritas Support website for more information:

http://www.veritas.com/docs/100022023

When setting up maintenance plans for databases within Availability Groups, the Agent job must take account of the local server replica state and must also be manually copied to all SQL Servers participating in the AG. For more information, see “SQL Agent and maintenance jobs” on page 45.

The plan should keep the database in good condition. However, these procedures are based upon a change ratio that, as the tables grow, can gradually degrade performance. Therefore, it is important to perform an additional inspection regularly to ensure that the databases are in optimal condition (fragmentation and statistics).

The following SQL statement can be used to identify all tables and indexes where the external fragmentation exceeds 5% and the table consists of at least 1,000 pages.

```
SELECT OBJECT_NAME(i.object_id) AS TableName,

i.name AS TableIndexName, phystat.avg_fragmentation_in_percent

FROM sys.dm_db_index_physical_stats(DB_ID(), NULL, NULL, NULL,
'LIMITED') phystat

JOIN sys.indexes i ON i.object_id = phystat.object_id AND
i.index_id = phystat.index_id

WHERE phystat.avg_fragmentation_in_percent > 5 AND
phystat.page_count > 1000
```

**Note**: Enterprise Vault 11.0.1 will execute a similar query on each Enterprise Vault database on a weekly basis to generate fragmentation warning reports in the Enterprise Vault system status and SCOM management pack.

Rebuilding indexes can be time-consuming, so low levels of fragmentation should be addressed with a reorganize instead. The following table recommends the most appropriate action to take according to the level of fragmentation:

| Fragmentation | Corrective approach |
|---|---|
| 5% – 30% | Reorganize |
| More than 30% | Rebuild (Can be achieved as online rebuild) |

The system stored procedure sp_updatestats (available in Maintenance Plans) should be executed on a regular basis to ensure all statistics are up to date. However, in some circumstances this may not update all desired statistics. The best way to perform these optimizations is through monitoring to identify any statistics which may require updating.

A method of identifying whether statistics should be manually updated using UPDATE STATISTICS might be to execute the following query:

```
SELECT object_name(i.object_id) as 'table_name',

       i.name as 'index_name',

       STATS_DATE(i.object_id, i.index_id) as
'last_stat_update',

       (case WHEN rowcnt>rowmodctr THEN abs((1-(cast(rowmodctr
AS float)/cast(rowcnt AS float)))*100) WHEN rowcnt<rowmodctr
THEN abs((1-(cast(rowcnt AS float)/cast(rowmodctr AS
float)))*100) ELSE 0.0 END) as 'percent_change'

FROM sys.indexes i

LEFT JOIN sys.objects o ON o.object_id = i.object_id

LEFT JOIN sys.sysindexes si ON o.object_id = si.id

WHERE o.type='U' AND rowmodctr>0 AND rowmodctr<rowcnt

AND (case WHEN rowcnt>rowmodctr THEN abs((1-(cast(rowmodctr AS
float)/cast(rowcnt AS float)))*100) WHEN rowcnt<rowmodctr THEN
abs((1-(cast(rowcnt AS float)/cast(rowmodctr AS float)))*100)
ELSE 0.0 END) > 5
```

**Note**: As of SQL 2005 the sysindexes view is provided for compatibility purposes, and the rowmodctr value is a calculated value which may not be accurate. However, it should still be sufficient to indicate outdated statistics.

For example, the following tables (and in particular their indexes or statistics) benefit from specific maintenance. These tables are very active and have a number of indexes to maintain.

| Database | Table | Notes |
|---|---|---|
| Directory | ACE | Used to store security descriptor references. Likely to have a higher read-to-write ratio. |

| Database | Table | Notes |
|---|---|---|
| Vault Store | Saveset | Used to store all archived items. Likely to have an equal or higher write-to-read ratio. |
| Vault Store | JournalArchive | Used during archiving process and as part of vault cache workflow. This table will have a very high turnover but will remain quite large. Likely to have an equal or higher write-to-read ratio. |
| Vault Store | SavesetContent | Used to store archived item metadata and metadata store data. |
| Vault Store | JournalStub | Used to store transient vault cache data |
| ADSS | Search | Used to store submitted search details |
| ADSS | ArchiveStatus | Used to record current search archive metrics |

The database file placement on disk can also lead to file fragmentation, which can degrade performance. If multiple database files reside on a single partition and tend to grow regularly, the database data and log file disks may need to be regularly file defragmented to maintain performance.

**Note**: The database maintenance plan should not include a data file shrink, to avoid unnecessary file growths. However, the database log files may need to be shrunk after backing up.

# Moving databases

It may be necessary to move the Enterprise Vault databases, for example to scale out to additional SQL servers, retire a SQL server or move to an availability group server and its listener. See the following article for guidelines on how to do this:

http://www.veritas.com/docs/100016654

Enterprise Vault 12 introduced two PowerShell tools that enable you to gather detailed Enterprise Vault database configuration information and reconfigure Enterprise Vault to connect to moved databases. See the Enterprise Vault *PowerShell Cmdlets Guide* for more information on Get-EVDatabaseDetail and Set-EVDatabaseDetail, which also includes instructions for moving databases into availability groups.

**Note**: The ADSS database location should be updated using the ADSS Configuration Wizard, which must be run on the ADSS middle tier server.

# Database upgrades

The version of SQL Server running Enterprise Vault databases may need to be upgraded. It is recommended the following steps are included in the upgrade:

- Perform a backup of all the Enterprise Vault databases.
- Perform the SQL Server upgrade.
- Check that the Enterprise Vault databases' compatibility level matches the current supported server version (maximum supported level 120).
- Change the Enterprise Vault databases' recovery models to "SIMPLE".
- Rebuild indexes.
- Update statistics with full scan.
- Return the recovery models to their original settings (recommended "FULL").
- Perform a backup of all the Enterprise Vault databases.

**Note**: Upgrading SQL Servers that are part of failover cluster instances or hosting availability groups requires additional planning to ensure the SQL environment remains consistent at all times. See Microsoft upgrade instructions and best practices described in in MSDN article "Upgrade and Update of Availability Group Server with Minimal Downtime and Data Loss".

Enterprise Vault product upgrades will most likely require upgrading the database schemas. The upgrade process may require significant additional storage capacity during the upgrade, most notably at the transaction logs and tempdb database. It is recommended the following steps are included in the upgrade:

- Perform a backup of all the Enterprise Vault databases.

- Temporarily increase the tempdb database storage or enable auto-growth.
- Change the Enterprise Vault databases' recovery models to "SIMPLE".
- Perform the Enterprise Vault database upgrades.
- Rebuild indexes.
- Update statistics.
- Return the recovery models to their original settings (recommended "FULL").
- Perform a backup of all the Enterprise Vault databases.

# Rolling over databases

As the Enterprise Vault environment grows, it may become necessary to roll over certain databases to remain within the available storage capacity and to maintain performance.

## Audit database

The auditing database can grow significantly in size depending upon the audit collection settings and the Enterprise Vault loads. The audit database storage requirements can be managed by either removing unwanted data, or rolling over to a new database. Before making any changes, it is recommended to make a backup of the EnterpriseVaultAudit database.

The AuditTrail table will grow significantly and can be managed by deleting unwanted rows. However, as this table can become extremely large, running a query to delete rows based upon their date will be very expensive and time consuming. Therefore, the best approach would be to back up and then truncate the existing table. See Veritas technical article, http://www.veritas.com/docs/100016653.

The audit database can be rolled over to a new database, which requires updating the ODBC Data Source Name on each Enterprise Vault server. See Veritas technical article http://www.veritas.com/docs/100016653.

## Vault store database

The number of vault store databases required should be determined during the Enterprise Vault solution design to meet the desired environment scalability. However, over time it may become necessary to roll over vault store databases to maintain databases within the recommended size limits.

**Note**: Rolling over a vault store partition does not roll over the vault store database.

Typically, the Vault Stores that may require rolling over will be journal-based Vault Stores which have a very high turnover of archived items. The approach to rolling over journal-based Vault Stores would be to create a new Vault Store and new archives then change the Enterprise Vault journal tasks to use the new archives.

**Note**: You cannot consolidate vault store databases.

Chapter 7

# Monitoring

Regular monitoring will enable a baseline performance profile to be measured and used for comparison over time to identify any trends in performance.

The SQL Server Data Collector can be used to gather and store performance metrics in order to monitor trends over time. Dynamic Management Views can also be used to provide current performance metrics.

Monitor the Enterprise Vault SQL servers during particular activities to ensure that the environment is performing correctly and allow appropriate database tuning.

Example activities to monitor for benchmark purposes are the following:

- During daytime activities which will include end-user workloads
- During mailbox archiving windows
- During journaling, typically daytime
- Overnight during activities such as mailbox or folder synchronization

Remember that, in isolation, these activities do not represent peak load. In production use, the database is also under load from combined activities including ad-hoc administrator actions.

You can also use Windows System Monitor to obtain system and SQL Server statistics. Typically, you should monitor the following counters.

| Object | Counters | Instances |
|---|---|---|
| PhysicalDisk (and potentially LogicalDisk as well) | Avg. Disk Read Queue Length<br>Avg. Disk Write Queue Length<br>Disk Transfers/sec<br>Avg. Disk Bytes/Transfer | SQL Data and log file drives |

| Object | Counters | Instances |
|---|---|---|
| | Avg. Disk sec/Transfer<br>Disk Bytes/sec<br>Split IO/sec | |
| Memory | Page Faults/sec<br>Pages/sec<br>Available Bytes | |
| Processor | % Processor Time<br>% Privileged Time<br>% Interrupt Time | _Total & All processors |
| System | Processor Queue Length<br>Context Switches/sec | |
| SQLServer:Buffer Manager | Buffer cache hit ratio<br>Page life expectancy<br>Procedure cache pages<br>Lazy writes/sec<br>Checkpoint pages/sec | |
| SQLServer:Access Methods | Page Splits/sec<br>Full Scans/sec | |
| SQLServer:Memory Manager | Total Server Memory (KB)<br>Target Server Memory (KB) | |
| SQLServer:Databases | Transactions/sec | _Total & Discovery Accelerator Database |
| SQLServer:SQL Statistics | Batch Requests/sec<br>SQL Compilations/sec<br>SQL Re-Compilations/sec | |
| SQLServer:Locks | Average Wait Time(ms)<br>Lock Timeouts/sec<br>Lock Waits/sec<br>Number of Deadlocks/sec | _Total |
| SQLServer:Latches | Average Latch Wait Time(ms) | |

| Object | Counters | Instances |
|---|---|---|
| | Latch Waits/sec | |

# CPU and memory

The % Processor Time for the _Total counter indicates overall system activity, but it may be worth monitoring the individual processor counters to see if any particular processors are heavily loaded for sustained periods.

If the % Processor Time is generally above 80%, and the Processor Queue length is generally above twice the number of allocated CPU cores, then the CPUs are likely to be a bottleneck. However, in SQL Server, high CPU use can be an indication of other factors such as ineffective indexes.

In addition, if the context switches/sec are above 15,000 per allocated CPU core when you experience high CPU, it is possible that the server is spending too much time switching between threads of equal priority (but only if the CPU time is above 80%). This may occur for various reasons, as described in the Microsoft books. However, this is most likely to occur with other co-existing software such as multiple SQL Server instances. In this situation, see the section “Multiple SQL Server instances” on page 38.

SQL Server should normally manage memory allocation automatically and avoid situations where memory paging can occur. However, it would be worth monitoring the memory counter Pages/sec, which records the number of hard faults to disk. If there are significant or sustained hard faults, trace the problem to the source. Watching the other SQL Server metrics listed below should also help to indicate if memory capacity is a bottleneck.

# Disk

Typically, the disk read/write queue length counters are monitored for evidence of a disk bottleneck. The queues should not normally rise above twice the number of disks in the respective arrays.

Monitor the average disk sec/transfer to measure the latency of the device accesses. Ideally, this should be approximately 5 - 15ms. However, anything in excess of 20ms is of concern.

The use of these counters may not be appropriate when using a SAN, and the hardware vendor’s tools should be used instead.

The Split IO/sec counter can indicate evidence of file fragmentation, and high levels should be addressed with file defragmentation. The remaining counters can be used to measure transfer rates and throughput.

**Note**: The physical disk counters represent all activity to a physical disk or array, which may contain several partitions (logical drives). The logical disk counters monitor activity to individual logical drives, so they can be used to identify which logical partition is utilizing the disks.

# SQL Server

You can monitor the SQL Server performance counters to indicate workload and performance problems. Typically, the following counters should be monitored.

| Counter | Notes |
|---|---|
| Buffer Cache Hit Ratio | Should be above 90% to avoid too much I/O. A lower value may indicate too little server memory. |
| Total Server Memory, Target Server Memory | If the total server memory is equal to the target server memory, there may be an issue with memory pressure. Examine the other SQL counters to help determine if the server may require more memory. |
| Page Life Expectancy | Indicates how long pages remain in memory. Values that are regularly less than 300 seconds may indicate insufficient memory. |
| Lazy Writes/sec | Indicates how many times the lazy writer is moving changed pages to disk to free buffer space. This should be quite low. High values indicate high I/O, which more memory will help to reduce. |
| Page splits/sec | Ideally should be around or below 80 – 100 per second. Index fill factors can be examined to improve this situation. |
| Batch Requests/sec, Transactions/sec(_Total) | Can indicate the number of SQL requests, and therefore the overall load the SQL Server is handling. |

As well as monitoring the system counters, you can extract more detailed information from SQL Server to identify potential performance issues and enable specific tuning.

You can measure the amount of time that is spent waiting for I/O operations using a SQL Server system table-valued function, fn_virtualfilestats. The following query displays the database files and the average time spent waiting on I/O (both read and write):

```
SELECT db_name(database_id),
file_id,io_stall/(num_of_reads+num_of_writes) as 'Avg IO wait
(ms)'

FROM sys.dm_io_virtual_file_stats(db_id('<database_name>'),
NULL)
```

Where `<database_name>` is the name of an Enterprise Vault database, or replace the db_id() with NULL to view all databases.

An average value above 20ms suggests that the I/O subsystem could be the source of a bottleneck.

---

**Note:** This displays an average since the database was created, and therefore any changes in hardware will not reflect an accurate change in this query. Instead, the io_stall column should be measured at intervals over several hours and the deltas used to determine improvements.

---

It is essential to measure the index fragmentation for particular key tables, as described in High Availability on page 41.

You can execute the following SQL statement, which outputs statistics on all tables and indexes where the external fragmentation exceeds 5% and the table consists of at least 1,000 pages:

```
SELECT OBJECT_NAME(i.object_id) AS TableName,

i.name AS TableIndexName, phystat.avg_fragmentation_in_percent

FROM sys.dm_db_index_physical_stats(DB_ID(), NULL, NULL, NULL,
'LIMITED') phystat

JOIN sys.indexes i with (nolock) ON i.object_id =
phystat.object_id AND i.index_id = phystat.index_id

WHERE phystat.avg_fragmentation_in_percent > 5 AND
phystat.page_count > 1000
```

---

**Note**: This query may take several minutes to complete, depending on the size of the database.

---

The following queries can be used on the Enterprise Vault databases to gather various metrics.

# Useful queries

The following queries may be useful in monitoring and tracking Enterprise Vault activity.

## Identifying vault store throughput

The Enterprise Vault Reporting and FSA Reporting features provide high-level storage and throughput statistics that you should use for normal reporting purposes. If you require high-level statistics on an ad-hoc basis, you can use the following queries on vault store databases to gather a summary of overall throughput over time.

The following query provides high-level hourly statistics for the past 24 hours:

```
DECLARE @start DATETIME = DATEADD(HOUR,-24,DATEADD(HOUR,
DATEDIFF(HOUR, 0,GETUTCDATE())-1, 0));

DECLARE @end DATETIME = DATEADD(HOUR, DATEDIFF(HOUR,
0,GETUTCDATE())+1, 0);


WITH gendate(ArchDate) AS

(

  SELECT TOP (DATEDIFF(HOUR, @start, @end) + 1) gendate =
DATEADD(HOUR, number-1, @start)

  FROM Numbers with (nolock) ORDER BY number

)

SELECT

        gendate.ArchDate as 'Archived Hour',

        COUNT(DISTINCT ArchivePointIdentity) as 'Target
Archives',

        COALESCE(SUM(TotalArchivedItems),0) as 'Hourly Rate',

        COALESCE(SUM(OriginalSize)/1048576,0) as 'MB (orig)',

        COALESCE(SUM(ArchivedItemSize)/1024,0) as 'MB (comp)'

FROM gendate LEFT OUTER JOIN ArchiveUsageTimeBasedSummary au
with (nolock)
```

```
        ON au.ArchivedDate >= @start AND au.ArchivedDate <= @end
AND gendate.ArchDate = DATEADD(HOUR, DATEDIFF(HOUR, 0,
au.ArchivedDate), 0)

        AND au.ConsolidationLevel=0

WHERE gendate.ArchDate >= @start AND gendate.ArchDate <= @end

GROUP BY gendate.ArchDate

ORDER BY gendate.ArchDate asc
```

The following query provides high-level daily statistics for the past 30 days:

```
DECLARE @start DATETIME = DATEADD(DAY,-30,DATEADD(DAY,
DATEDIFF(DAY, 0,GETUTCDATE())-1, 0));

DECLARE @end DATETIME = DATEADD(DAY, DATEDIFF(DAY,
0,GETUTCDATE())+1, 0);


WITH gendate(ArchDate) AS

(

  SELECT TOP (DATEDIFF(DAY, @start, @end) + 1) gendate =
DATEADD(DAY, number-1, @start)

  FROM Numbers with (nolock) ORDER BY number

)

SELECT

        gendate.ArchDate as 'Archived Day',

        COUNT(DISTINCT ArchivePointIdentity) as 'Target
Archives',

        COALESCE(SUM(TotalArchivedItems),0) as 'Daily Items',

        COALESCE(SUM(OriginalSize)/1048576,0) as 'MB (orig)',

        COALESCE(SUM(ArchivedItemSize)/1024,0) as 'MB (comp)'

FROM gendate LEFT OUTER JOIN ArchiveUsageTimeBasedSummary au
with (nolock)

        ON au.ArchivedDate >= @start AND au.ArchivedDate <= @end
AND gendate.ArchDate = DATEADD(DAY, DATEDIFF(DAY, 0,
au.ArchivedDate), 0)
```

```
        AND au.ConsolidationLevel=0
WHERE gendate.ArchDate >= @start AND gendate.ArchDate <= @end
GROUP BY gendate.ArchDate
ORDER BY gendate.ArchDate asc
```

Note the following:

- The summary tables retain statistics for a limited period, so not all time periods are available.
- You can adjust the time period by changing the `start` and `end variables`. For example, consider the following lines:

  ```
  DECLARE @start DATETIME = dateadd(…)
  DECLARE @end DATETIME = dateadd(…)
  ```

- You can change this to a specific date range by replacing the GETUTCDATE with specific dates:

  ```
  DECLARE @start DATETIME = DATEADD(HOUR,-1,DATEADD(HOUR,
  DATEDIFF(HOUR, 0, '2015-11-27 12:00:00'), 0));
  DECLARE @end DATETIME = DATEADD(HOUR, DATEDIFF(HOUR, 0,
  '2015-11-27 23:00:00')+1, 0);
  ```

Where necessary, you can use the following queries to examine the data in more detail.

The following query lets you examine the rate at which items were added to the Vault Store storage queue, together with the original and compressed data sizes, for the past 24 hours. The query does not directly show the rate at which items were archived or indexed, however the storage queue will be constrained by the archiving rate.

```
DECLARE @start DATETIME = DATEADD(HOUR,-24,DATEADD(HOUR,
DATEDIFF(HOUR, 0,GETUTCDATE())-1, 0));
DECLARE @end DATETIME = DATEADD(HOUR, DATEDIFF(HOUR,
0,GETUTCDATE())+1, 0);


WITH gendate(ArchDate) AS
(
  SELECT TOP (DATEDIFF(HOUR, @start, @end) + 1) gendate =
DATEADD(HOUR, number-1, @start)
```

```
  FROM Numbers with (nolock) ORDER BY number

)

SELECT

  gendate.ArchDate,

  "Items Archived" = count(ss.SavesetIdentity),

  "MB (original)" = COALESCE(sum(cast(sp.originalsize as
float))/1024/1024,0),

  "MB (compressed)" = COALESCE(sum(cast(ss.itemsize as
float))/1024,0)

FROM gendate LEFT OUTER JOIN saveset ss with (nolock)

        ON ss.ArchivedDate >= @start AND ss.ArchivedDate <= @end
AND gendate.ArchDate = DATEADD(HOUR, DATEDIFF(HOUR, 0,
ss.ArchivedDate), 0)

    left outer join savesetproperty sp with (nolock)

    on ss.savesetidentity = sp.savesetidentity

WHERE gendate.ArchDate >= @start AND gendate.ArchDate <= @end

GROUP BY gendate.ArchDate

ORDER BY gendate.ArchDate asc
```

You can examine the rate at which items are archived while items are still active in the journal records. These are stored for 32 days after archiving and removed if the items have been indexed and marked as secure.

```
DECLARE @start DATETIME = DATEADD(HOUR,-24,DATEADD(HOUR,
DATEDIFF(HOUR, 0,GETUTCDATE())-1, 0));

DECLARE @end DATETIME = DATEADD(HOUR, DATEDIFF(HOUR,
0,GETUTCDATE())+1, 0);

WITH gendate(ArchDate) AS

(

  SELECT TOP (DATEDIFF(HOUR, @start, @end) + 1) gendate =
DATEADD(HOUR, number-1, @start)

  FROM Numbers with (nolock) ORDER BY number

)
```

```
SELECT

  "Archived Date" = gendate.ArchDate,

  "Items Archived" = count(ja.SavesetIdentity),

  "MB (original)" = COALESCE(sum(cast(sp.originalsize as
float))/1024/1024,0),

  "MB (compressed)" = COALESCE(sum(cast(ss.itemsize as
float))/1024,0)

FROM gendate LEFT OUTER JOIN journalarchive ja with (nolock)

       ON ja.AsyncIngestionCompletionTime >= @start AND
ja.AsyncIngestionCompletionTime <= @end AND gendate.ArchDate =
DATEADD(HOUR, DATEDIFF(HOUR, 0,
ja.AsyncIngestionCompletionTime), 0)

    left outer join Saveset ss with (nolock)

    on ja.SavesetIdentity = ss.SavesetIdentity

    left outer join savesetproperty sp with (nolock)

    on ja.savesetidentity = sp.savesetidentity

WHERE gendate.ArchDate >= @start AND gendate.ArchDate <= @end

GROUP BY gendate.ArchDate

ORDER BY gendate.ArchDate asc
```

Note the following:

- Depending on the size of the database, this query can take some time to complete. It may significantly affect performance while it is running.
- You can adjust the time period by changing the `start and end variables`. For example, consider the following lines:

  ```
  DECLARE @start DATETIME = dateadd(…)

  DECLARE @end DATETIME = dateadd(…)
  ```

  You can change this to a specific date range, by replacing the GETUTCDATE with specific dates:

  ```
  DECLARE @start DATETIME = DATEADD(HOUR,-1,DATEADD(HOUR,
  DATEDIFF(HOUR, 0, '2015-11-27 12:00:00'), 0));

  DECLARE @end DATETIME = DATEADD(HOUR, DATEDIFF(HOUR, 0,
  '2015-11-27 23:00:00')+1, 0);
  ```

- Avoid executing the query during heavy load, particularly during archiving or journal archiving periods.

The following is sample output from the query:

| Archived date | Items archived | MB (original) | MB (compressed) |
|---|---|---|---|
| 2016-02-08 07:00 | 106753 | 10483 | 7830 |
| 2016-02-08 08:00 | 107506 | 10470 | 7857 |
| 2016-02-08 09:00 | 107150 | 10554 | 7835 |

## Identifying sharing group sharing ratio

Execute the following query against each fingerprint database. It outputs the number of sharable parts stored, the number of references to parts from archived items and the SIS ratio (average references per stored part).

```
DECLARE @sCmd nvarchar(1000)

DECLARE @curTable  int, @LastTable int DECLARE @tbls TABLE
(rownum int IDENTITY(1,1) NOT NULL, TableName nvarchar(255) NOT
NULL,OwnerName nvarchar(255) NOT NULL)

CREATE TABLE #Results (dcount float, drefcount float)

INSERT @tbls (TableName, OwnerName) SELECT object_name(o.id),
u.[name] FROM sysobjects o INNER JOIN sysusers u ON u.uid =
o.uid WHERE o.type='U' and object_name(o.id) like
'MemberTable____'

SET @LastTable = @@ROWCOUNT

SET @curTable = 0

WHILE (@curTable < @LastTable)

BEGIN   SET @curTable = @curTable + 1
```

```
            SELECT @sCmd = N'SELECT cast(count(*) as float) as
''dcount'',cast(sum(refcount) as float) as ''drefcount'' FROM '
+ LTRIM(RTRIM(OwnerName)) + N'.' + TableName + N' with
(nolock)'

                        FROM @tbls

                        WHERE rownum = @curTable

            INSERT #Results EXEC sp_executesql @sCmd

END

SELECT SUM(dcount) as 'parts_stored',SUM(drefcount) as
'part_references',SUM(drefcount)/SUM(dcount) as 'SIS_ratio'
FROM #Results

DROP TABLE #Results

GO
```

Note the following:

- Depending upon the size of the database this query may take some time to complete.
- Avoid executing the query during heavy load, particularly during archiving or journal archiving periods.

The following is sample output from the query.

| parts_stored | part_references | SIS_ratio |
|---|---|---|
| 484583 | 5414621 | 11.173 |