Estimate performance and capacity requirements for Microsoft Search Server 2010 Express

Estimate performance and capacity requirements for Microsoft Search Server 2010 Express

Brion Stone
Microsoft Corporation

April 2010

Applies to: Microsoft Search Server 2010 Express

Summary: This document provides capacity planning information for Microsoft® Search Server 2010 Express. It includes the following information for a test configuration:

- Test environment specifications, such as hardware, farm topology, and configuration
- The workload used for data generation, including the number and class of users, and farm usage characteristics
- Test farm dataset, including database contents, search indexes and external data sources
- Health and performance data specific to the tested environment
- Test data and recommendations for how to determine the hardware and configuration you need to deploy a similar environment, and how to optimize the environment for appropriate capacity and performance characteristics

Contents

Contents

Introduction

This document describes a collaboration environment deployment of Search Server 2010 Express. Information in this document includes:

- Test environment specifications, such as hardware, farm topology, and configuration
- The workload used for data generation, including the number and class of users, and farm usage characteristics
- Test farm dataset, including database contents and sizes, search indexes, and external data sources
- Health and performance data specific to the tested environment

It also contains common test data and recommendations for how to determine the hardware, topology and configuration that you need to deploy a similar environment, and how to optimize the environment for appropriate capacity and performance characteristics.

Search Server 2010 Express contains a richer set of features than earlier versions. Before you employ this architecture to deliver more powerful features and functionality to your users, you must carefully consider the impact upon the farm's capacity and performance.

When you read this document, you will understand how to:

- Define performance and capacity targets for your environment.
- Plan the hardware required to support the number and type of users, and the features you intend to deploy.
- Design the physical and logical topology for optimum reliability and efficiency.
- Test, validate, and scale the environment to achieve performance and capacity targets.
- Monitor the environment for key indicators.

Before you read this document, you should read the following:

- Capacity management and sizing for SharePoint Server 2010
- SharePoint Server 2010 Capacity Management: Software Boundaries and Limits
- Database-specific content

Search Life Cycle

This document allows you to estimate capacity at an early stage of the farm. Farms move through multiple stages as content is crawled:

- **Index acquisition** This is the first stage of data population. The duration of this stage depends on the size of the content repositories. It is characterized by:
  - Full crawls (possibly concurrent) of content.
  - Close monitoring of the crawl system to ensure that hosts being crawled are not a bottleneck for the crawl.
  - Frequent *master merges* that, for each query component, are triggered when a certain amount of the index has changed.
- **Index maintenance** This is the most common stage of a farm. It is characterized by:
  - Incremental crawls of all content.
  - For SharePoint content crawls, a majority of the changes encountered during the crawl are security changes.

    - Infrequent master merges that, for each query component, are triggered when a certain amount of the index has changed.
- **Index cleanup** This stage occurs when a content change moves the farm out of the index maintenance stage; for example, when a content database or site is moved from one search service application to another. This stage is triggered when:
    - A content source or start address is deleted from a search service application.
    - A host supplying content is not found by the search crawler for an extended period of time.

Scenario

This Search Server 2010 Express scenario describes a typical search deployment with the following characteristics:

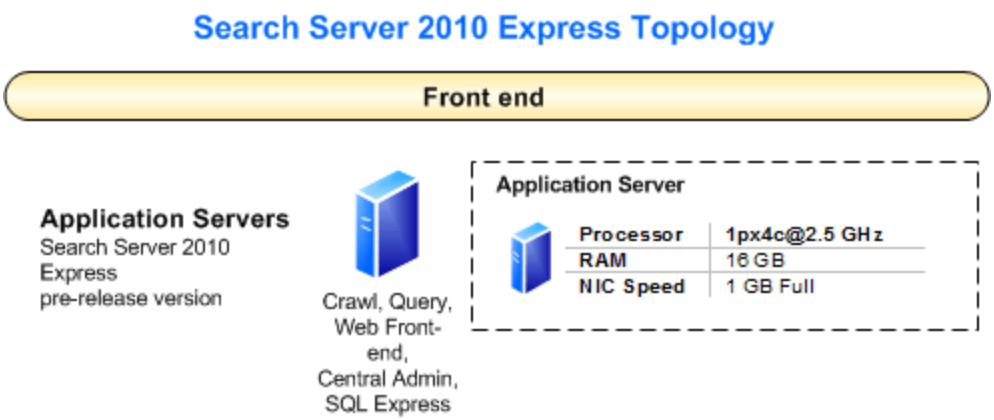
- Search is provided over a small amount of content. The initial 340,000 items in the index is typical for a small corpus. The second set of tests in the "Test Data" section was run at 500,000 items. Adding more items to this farm was not possible due to the 4 GB database size limit imposed by SQL Server® Express. We would need to upgrade our SQL Server edition in order to add more items to this farm, up to the item limit.
- Search can only be run on one server in the farm at a time. Scaling out is not supported.
- The vast majority of the user queries can be found in the same 33% of the index. This means that most users query for the same terms.
- The default metadata settings are used, ensuring that the property databases do not grow at a large rate.
- Measurements taken on these farms may vary due to network and environmental conditions. We can expect up to a 10% margin of error.
- The farm is not run as an evaluation. Because 16 gigabytes (GB) of RAM exists on the server, this configuration is meant for production use. Subsequently, the **PerformanceLevel** of the farm is set to **PartlyReduced** using by using the following Windows® PowerShell™ command:
    - Get-SPEnterpriseSearchService | Set-SPEnterpriseSearchService –PerformanceLevel "PartlyReduced"

General guidelines follow in the Recommendations section.

Specifications

This section provides detailed information about the hardware, software, topology, and configuration of the test environment.

Topology



Hardware

Note
Because the farm is running pre-release versions of Search Server 2010 Express, and the team wanted to avoid potential problems, the hardware used has more capacity than is required under more normal circumstances.

Application Servers

Server	Combined
Processor(s)	1px4c@2.5 GHz
RAM	16 GB
Operating system	Windows Server 2008 R1 SP2, 64-bit
Storage	2 x 147 GB 15K SAS: RAID1:OS 2 x 147 GB 15K SAS: RAID1:SQL Data, Index 2 x 147 GB 15K SAS: RAID1:SQL Logs
Number of NICs	1
NIC Speed	1 Gigabit
Authentication	NTLM
Load balancer type	None
Software version	Search Server 2010 Express (pre-release version)

Services running locally	All services (including SQL Server Express)

Workload

This section describes the workload used for data generation, including the number of users and farm usage characteristics.

Workload Characteristics	Value
High-level description of workload (collab, wcm, social…)	Search farms
Average queries per minute	6
Average concurrent users	1
Total queries per day	8640
Timer job load	Search Health Monitoring – Trace Events; Crawl Log Report; Health Analysis Job; Search and Process

Dataset

This section describes the test farm dataset, including database contents and sizes, search indexes, and external data sources.

Object	Value
Search index size (# of items)	334467
Sum (size of crawl database)	1.97 GB
Sum (size of crawl database log file)	.47 GB
Sum (size of property database)	2.54 GB
Sum (size of property database log file)	.149 GB
Size of search administration database	.076 GB
Sum (size of active index partitions)	1.98 GB
Total # of databases	4
Other databases	SharePoint Config; SharePoint AdminContent; State Service Database; BDC Application; WSS_UsageApplication; WSS_Content

Health and Performance Data

This section provides health and performance data specific to the test environment.

Query Performance Data

The following measurements were taken with 340K items in the search index. The columns give the measurements taken during the specific test, and the results are at the bottom of the table.

	Scorecard Metric	Query Testing
CPU Metrics	Avg (WFE/APP/SQL)CPU	99.065%
Reliability	Failure rate	Not performed
	Crashes	Not performed
Perf Counters	Cache Hit Ratio (SQL)	99.97%
	SQL Locks: Average Wait Time [ms]	0
	SQL Locks: Lock Wait Time [ms]	0
	SQL Locks: Deadlocks/s	0
	SQL Latches: Average Wait Time [ms]	2.983
	SQL Compilations/sec	64.192
	SQL Statistics: SQL Re-Compilations/s	.032
	Avg Disk queue length (WFE/APP/SQL)	.23
	Disk Queue Length: Writes (WFE/APP/SQL)	.013
	Disk Reads/sec (WFE/APP/SQL)	12.93
	Disk Writes/sec (WFE/APP/SQL)	108.7
	Average memory used (WFE/APP/SQL)	22.182%
	Max memory used (WFE/APP/SQL)	22.325%
Test Results	# Errors	0
	Query UI Latency (75th Percentile)	0.256 sec
	Query UI Latency (95th Percentile)	0.303 sec
	Query Throughput	14.743 Requests/sec

Crawl Performance Data

The following measurements were taken during full crawls of the given content repository (content repository size is given in parenthesis). The columns give the measurements taken during the specific crawl, and the results are at the bottom of the table.

	Scorecard Metric	SharePoint Foundation (140K)
CPU Metrics	Avg (WFE/APP/SQL)CPU	56.91%
Reliability	Failure rate	Not performed
	Crashes	Not performed
Perf Counters	Cache Hit Ratio (SQL)	98.57%
	SQL Locks: Average Wait Time [ms]	146.75
	SQL Locks: Lock Wait Time [ms]	201.85
	SQL Locks: Deadlocks/sec	0.002
	SQL Latches: Average Wait Time [ms]	13.747
	SQL Compilations/sec	8.161
	SQL Statistics: SQL Re-Compilations/s	0.253
	Avg Disk queue length (WFE/APP/SQL)	9.752
	Disk Queue Length: Writes (WFE/APP/SQL)	3.458
	Disk Reads/sec (WFE/APP/SQL)	362.94
	Disk Writes/sec (WFE/APP/SQL)	184.37
	Average memory used (WFE/APP/SQL)	27.37
	Max memory used (WFE/APP/SQL)	28.62

	Scorecard Metric	File Share (100K)	HTTP (100K)	SharePoint Foundation (140K)
Test Results	# Successes	100105	100000	138595
	# Errors	79	0	12
	Portal Crawl Speed	65.557	53.419	88.165

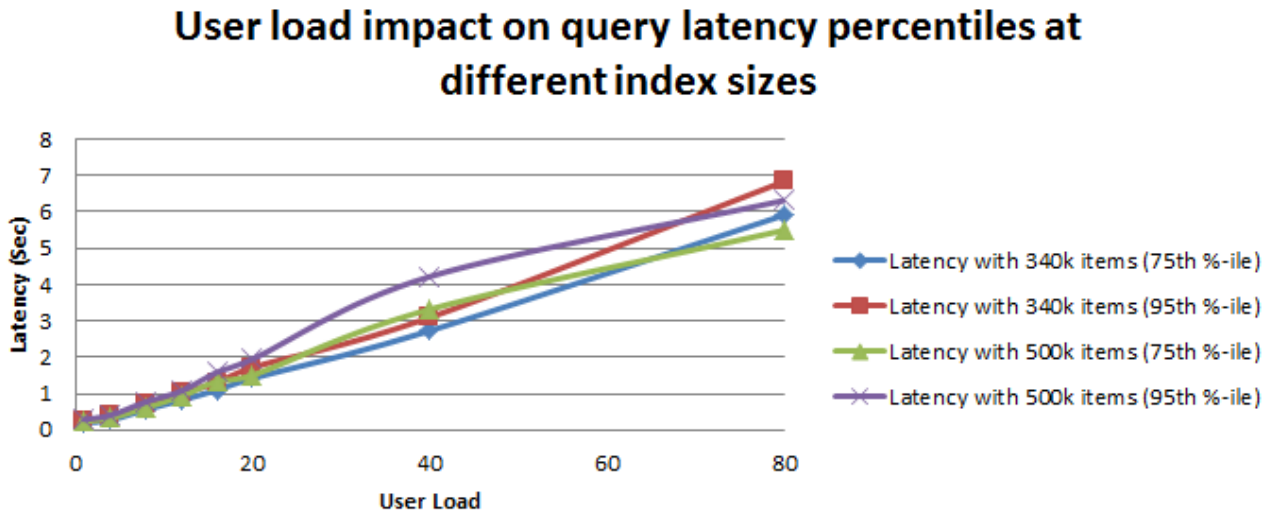
	(items/sec)			
	Anchor Crawl Speed (items/sec)	770.031	769.231	1108.76
	Total Crawl Speed (items/sec)	60.413	49.95	81.671

Test Data

This section provides test data illustrating how the farm performed under load. Refer to the Optimizations section to understand how to improve farm performance.

Query Latency

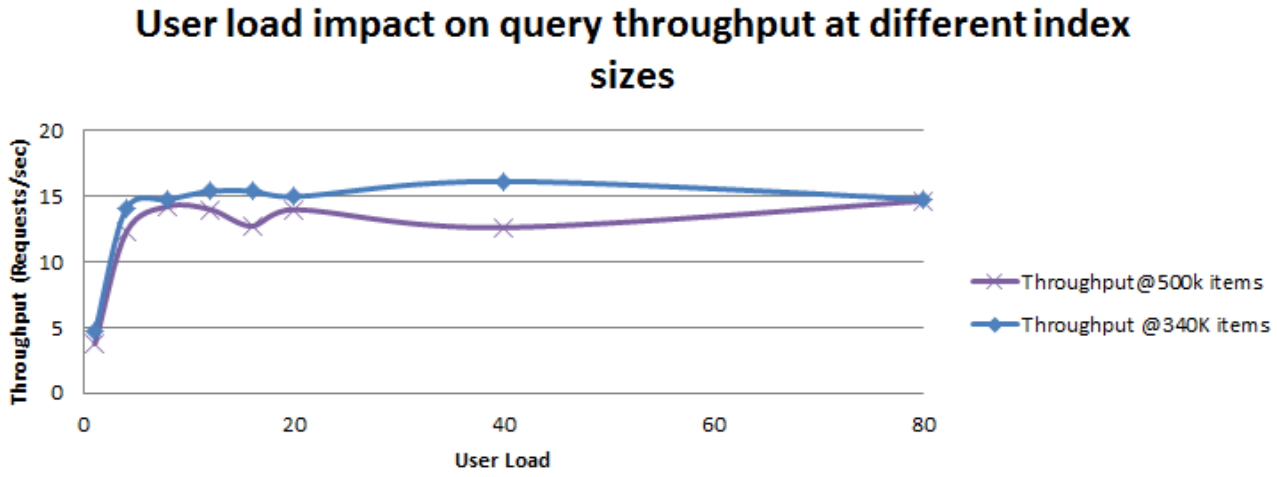
The following graph displays query latency percentiles for this farm as user load increases (gathered during the Query Throughput test). A query percentile of 95% means that 95% of the query latencies measured were below that value.



Takeaway: From this graph you can see that there is not much difference at light loads between 340K and 500K in the index.

Query Throughput

The following graph displays the query throughput for this farm as user load increases (gathered during the Query Throughput test).

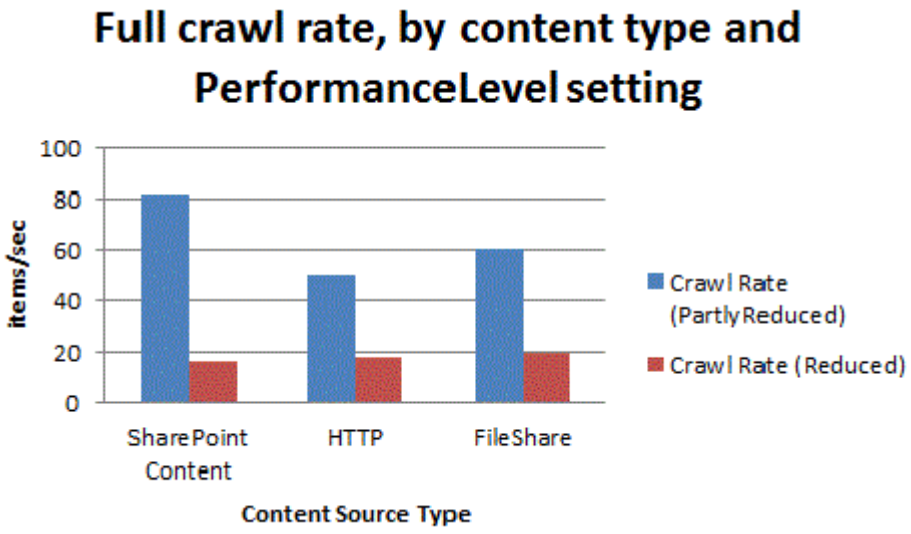


Takeaway: Taking into account both this graph and the last graph, you can see that adding additional user load beyond about eight concurrent users, this farm will get no additional throughput, and the user latencies will increase.

Crawl Rate

The following graph displays the crawl rate for this farm during the index acquisition stage of the search life cycle. The values represent crawl rates from full crawls of the same content sources, in items crawled per second. The two measurements are taken with the same farm; the only difference is that the second run changed the **PerformanceLevel** from **Reduced** to **PartlyReduced** by using the following Windows PowerShell cmdlets:

- Get-SPEnterpriseSearchService | Set-SPEnterpriseSearchService –PerformanceLevel "PartlyReduced"



Takeaway: By default, **PerformanceLevel** is set to **Reduced** in Search Server 2010 Express in order to satisfy the evaluation scenario where it is likely running on a server with minimal resources available. Setting the **PerformanceLevel** to **PartlyReduced** resulted in a large performance improvement because the server had the recommended resources available.

Overall Takeaway

During the query tests, the CPU was near capacity on the query servers. Adding more CPU cores to the server would be the first step in improving performance.

Also, note that the property database hit the 4 GB database limitation of SQL Server Express at just over 500,000 items in the index. Moving to another edition of SQL Server (without the 4 GB limit) would be the first step in scaling up the farm to the eventual 10 million item limit. Subsequent steps would include scaling up the hardware (more processors, RAM, faster storage, etc.).

Recommendations and Troubleshooting

This section provides recommendations for how to determine the hardware, topology, and configuration you need to deploy a similar environment, and how to optimize the environment for appropriate capacity and performance characteristics.

Recommendations

This section describes specific actions you can take to optimize your environment for appropriate capacity and performance characteristics.

Hardware recommendations

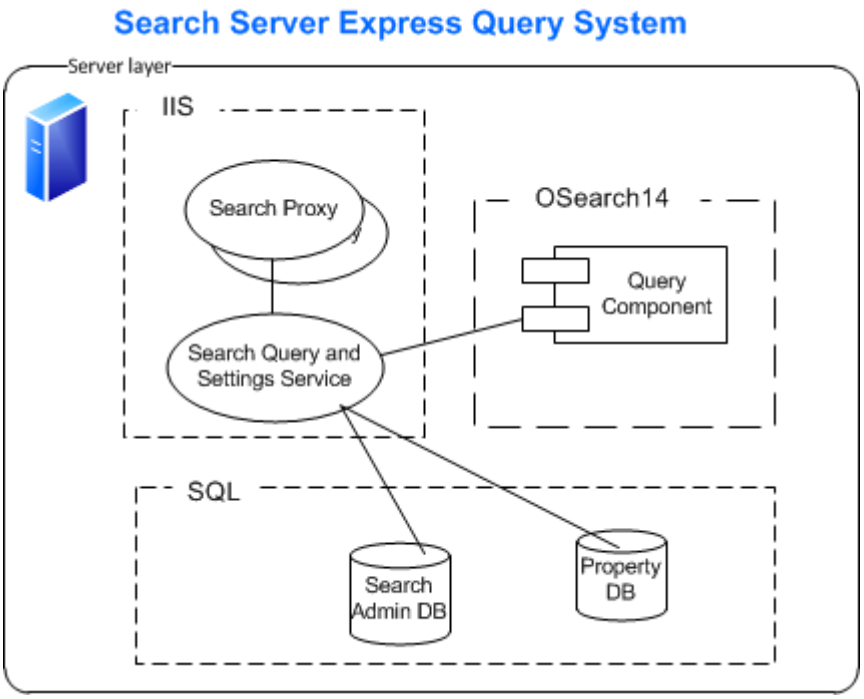
For specific information about overall minimum and recommended system requirements, see Determine hardware and software requirements. Note that requirements for servers used for search supersede those overall system requirements. Follow the recommended guidelines below for RAM, processor, and IOPS, in order to meet performance goals.

Search Sizing

This section explains the search system, including sizing requirements and guidelines, per component.

Search Query System

This section shows the components of the search query system for a given Search service application (SSA). The sizing requirements for each appear in the table below the diagram.



Object Descriptions

This section defines the search query system objects in the above diagram:

- **Search proxy** This is the SSA proxy that installs on any farm that consumes search from this SSA. It runs in the context of the Web applications that are associated with the SSA proxy.
- **Search Query and Site Settings Service** This is also known as the query processor (QP). Receiving the query from an SSA proxy connection, a QP does the following:
    - Sends the query to one active query component for each partition (and/or to the property database, depending on the query).
    - Retrieves Best Bets and removes duplicates to get the results set.
    - Security-trims the results based on security descriptors in the Search Administration database.
    - Retrieves the final results set's metadata from the property database.
    - Sends the query results back to the proxy.
- **Index partition** This is a logical concept that represents the full-text index; however, note that query component contains the index.
- **Search query component** A query component contains the full-text index. When queried by a QP, the query component determines the best results from its index and returns those items.
- **Search Administration database** Created at the same time as the SSA, the Search Administration database contains the SSA-wide data used for queries like Best Bets and security descriptors, as well as application settings used for administration.
- **Property database** A property database contains the metadata (title, author, related fields) for the items in the index. The property database is used for property-based queries as well as retrieving metadata needed for display of the final results.

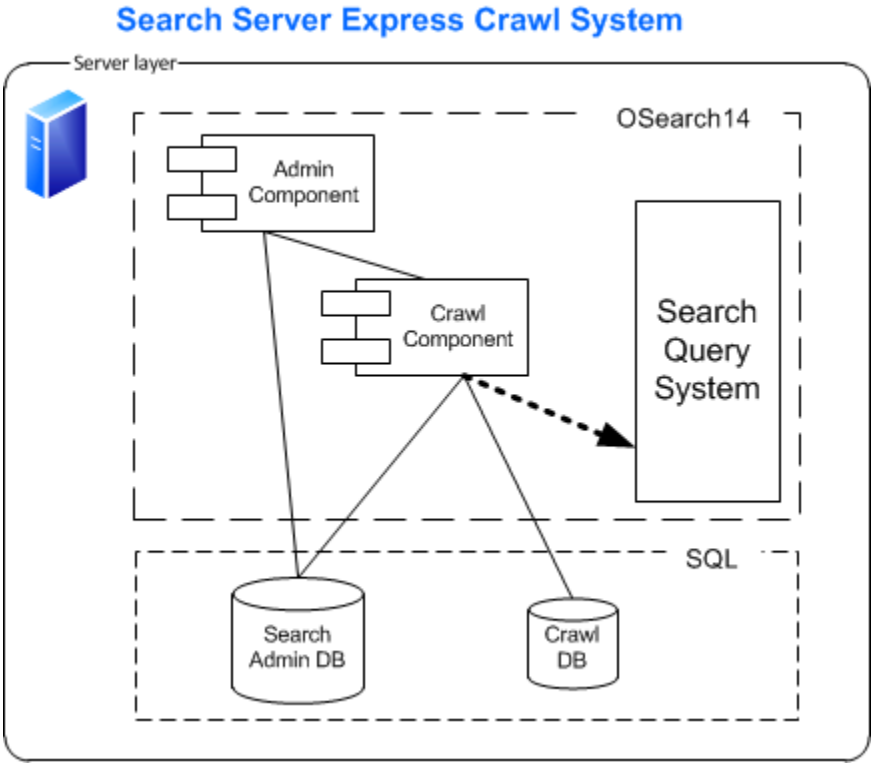
Scaling Details

Object	Scale Considerations	RAM	IOPS (read/write)
Search proxy	This scales with the Web front-end servers on which it is associated.	N/A	N/A
Search Query and Site Settings Service	This service, installed in the **Services on Server** page in Central Administration, should be started on the server. If a custom security trimmer is used, it may affect CPU and RAM resources.	This uses RAM (process cache) for caching security descriptors for the index.	N/A

Index partition	N/A	N/A	N/A
Query component	The query component on a server consumes memory when serving queries. It consumes IO when crawling is occurring.	A query component should have 33% of its index in RAM (OS cache).	1000 IOPS needed for: • Loading the index into RAM for queries. • Writing index fragments received from each crawl component. • Merging index fragments into its index, such as during a master merge.
Search Administration database	For each query, Best Bets and security descriptors are loaded from the Search Administration database. Ensure the database server has enough RAM to serve this from cache.	Ensure the database server has enough RAM to keep the critical table (MSSSecurityDescriptors) in RAM.	700 IOPS
Property database	For each query, metadata is retrieved from the property database for the document results, so scaling up the database server's RAM is a scale consideration.	Ensure the database server has enough RAM to keep 33% of the critical tables (MSSDocSDIDs + MSSDocProps + MSSDocresults) in cache.	2000 IOPS 30% read, 70% write

Search Crawl System

This section shows the components of the search crawl system. The sizing requirements of each appear in the table below the diagram.



Object Descriptions

This section defines the search crawl system objects in the above diagram:

- **Administration component** An administration component is used when starting a crawl, as well as when performing an administration task on the crawl system.
- **Crawl component** A crawl component processes crawls of content sources, propagates the resulting index fragment files to query components, and adds information about the location and crawl schedule of content sources to its associated crawl database.
- **Search Administration database** Created at the same time as the SSA, the Search Administration database stores the security descriptors discovered during the crawl, as well as application settings used for administration.
- **Crawl database** A crawl database contains data related to the location of content sources, crawl schedules, and other information specific to crawl operations. They can be dedicated to specific hosts by creating host distribution rules. A crawl database only stores data; the crawl component associated with the given crawl database does the crawling.
- **Search query system**

Scaling Details

Object	Scale Considerations	RAM	IOPS (Optionally, % read/write)
Administration component	The single administration component is not scalable.	Minimal	Minimal
Crawl component	Crawl components aggressively use CPU bandwidth. Optimally, a given crawl component can utilize four CPU cores. RAM is not as critical.	Moderate. Note that when crawling East Asian documents, RAM requirements will increase due to the word breakers.	300-400
Search Administration database	See query system table entry above.	See query system table entry above.	700
Crawl database	A crawl database aggressively uses IO bandwidth. RAM is not as critical. A crawl database needs 3.5K IOPS for crawling activities; it will consume as much as 6K IOPS, based on the available bandwidth.	Moderate	3500 – 6000 73% read, 27% write

Software limits

The following table gives software boundaries imposed to support an acceptable search experience:

Object	Limit	Additional Notes
SharePoint Search Application (SSA)	Does not apply to Search Server 2010 Express configuration.	
Indexed documents	Using SQL Server Express, this is limited to about 500K items. Otherwise, the overall recommended maximum is 10 million	SQL Server Express imposes a 4 GB database size limit, which translates to about 500K items in the search corpus. Moving to a different SQL Server edition would eliminate this limitation.

	items.	
Index Partition	Does not apply to Search Server 2010 Express.	
Property Database	Does not apply to Search Server 2010 Express.	
Crawl Databases	Does not apply to Search Server 2010 Express.	
Crawl Components	Does not apply to Search Server 2010 Express.	
Query Components	Does not apply to Search Server 2010 Express.	
Concurrent Crawls	Recommended limit is 2 per SSA.	This is the number of crawls underway at the same time. Crawls are extremely expensive search tasks which can impact database as well as other application load; exceeding 2 concurrent crawls may cause the overall crawl rate to degrade.
Content sources	Recommended limit of 50 content sources.	The recommended limit can be exceeded up to the hard limit of 500 per SSA. However, fewer start addresses should be used, and the concurrent crawl limit needs to be followed.
Start Addresses	Recommended limit of 100 start addresses per content source.	The recommended limit can be exceeded up to the hard limit of 500 per content source. However, fewer content sources should be used. A better approach when you have many start address is to put them as links on an HTML page, and then have the HTTP crawler crawl the page, following the links.
Crawl rules	Recommended limit of 100	The recommendation can be exceeded. However, display of the crawl rules on the Search Administration pages is degraded.
Crawl logs	Recommended limit of 10 million per application	This is the number of individual log entries in the crawl log. It will follow the "indexed documents" limit.
Metadata properties recognized per item	The hard limit is 10000.	This is the number of metadata properties that, when an item is crawled, can be determined (and potentially mapped/used for queries).
Crawled properties	500,000	These are properties that are discovered during a crawl.

Managed properties	100,000	These are properties used by the search system in queries. Crawled properties are mapped to managed properties. We recommend a maximum of 100 mappings per managed property. Exceeding this limit may degrade crawl speed and query performance.
Scopes	Recommended limit of 200 per site	This is a recommended limit per site. Exceeding this limit may degrade the crawl efficiency as well as impacting end-user browser latency if the scopes are added to the display group. Also, display of the scopes in the Search Administration pages degrades as the number of scopes increases past the recommended limit.
Display groups	25 per site	These are used for a grouped display of scopes through the user interface. Exceeding this limit will start degrading the Search Administration scope experience.
Scope Rules	Recommended limit is 100 scope rules per scope, and 600 total per SSA	Exceeding this limit will degrade crawl freshness and delay potential results from scoped queries.
Keywords	Recommended limit of 200 per site collection	The recommended limit can be exceeded up to the maximum (ASP.NET-imposed) limit of 5000 per site collection with five Best Bets per keyword. Display of keywords on the site administration user interface will degrade. The ASP.NET-imposed limit can be modified by editing the Web.Config and Client.config files (MaxItemsInObjectGraph).
Authoritative pages	Recommended limit of one top-level authoritative page, and as few as possible 2nd and 3rd level pages, while achieving desired relevance.	The hard limit is 200 per relevance level per SSA, but adding additional pages may not achieve the desired relevance. Add the key site to the 1st relevance level. Add subsequent key sites either 2nd or 3rd relevance levels, one at a time, evaluating relevance after each addition to ensure that the desired relevance effect is achieved.
Alerts	Recommended limit of 1,000,000 per SSA	
Results removal	100 URLS in one operation	This is the maximum recommended number of URLs that should be removed from the system in one operation.

Optimizations

This following sections discuss methods for improving farm performance.

Search Crawl System Optimizations

In general, search crawl optimizations follow one of the following scenarios: users complain about query results that either should be there but aren't, or are there but are stale.

While attempting to crawl the content source start address within freshness goals, you may run into the following crawl performance issues:

- Crawl rate is low due to IOPS bottlenecks in the search crawl database subsystem.
- Crawl rate is low due to lack of CPU threads in the search crawl subsystem.
- Crawl rate is low due to slow repository responsiveness.

Each of these issues assumes that the crawl rate is low. Use the search administration reports to create a typical baseline crawl rate for your system over time. The following sub-sections describe various ways of addressing crawl performance issues when this baseline regresses.

Crawl IOPS bottleneck

After determining that a crawl or property database is a bottleneck, you need to scale out the crawl system to address it using the appropriate resolutions. Always check the crawl database to make sure it is not the bottleneck. If crawl database IOPS are already bottlenecked, increasing the number of threads does not help.

Crawl CPU thread bottleneck

If you have a large number of hosts, and have no other crawl bottlenecks, you need to scale up or scale out your crawl system to address it using the appropriate resolutions. The crawler can accommodate a maximum of 256 threads per search service application. We recommend having a quad core processor to realize the full benefit of the maximum number of threads. Once it is conclusively determined that the repository is serving data fast enough (see Crawl bottleneck on repository in the following section), the crawl throughput can be increased by requesting data faster from the repository by increasing the number of crawler threads. This can be achieved in two ways as outlined below:

1. Change the indexer performance level to **Partially Reduced** or **Maximum** by using the following Windows PowerShell command. The **Maximum** value is used if using a processor with fewer than four cores.
    a. Get-SPEnterpriseSearchService | Set-SPEnterpriseSearchService –PerformanceLevel "Maximum"
2. Use crawler impact rules to increase the number of threads per host. This should take into consideration that the system supports a maximum of 256 threads, and assigning a large number of threads to a few hosts might result in slower data retrieval from other repositories.

Crawl bottleneck on repository

At times, when crawling a SharePoint Web application with many nested site collections or remote file shares, the search crawler might be bottlenecked on the repository. A repository bottleneck can be identified if the following two conditions are true:

1. There is a low (<20%) CPU utilization on the server that is running Search Server 2010 Express.
2. There is a large number of threads (almost all in the worst case) waiting on the network.

    This is identified by looking at OSS Search Gatherer/Threads Accessing Network performance counter.

This means that the threads are blocked waiting for data from the repository. In an environment with multiple content sources, it might be useful to identify the host whose responsiveness is slow by pausing all other crawls and just crawling the content source having the suspected host as one of its start address.

Once we've identified a problematic host, we need to investigate the cause of their slow response times. For SharePoint content in particular, see Capacity Planning and Sizing for Microsoft SharePoint 2010 Divisional Portal, especially adding additional front-end Web servers which search can use for crawling.

> Takeaway: The crawl throughput can be significantly improved by performance tuning the crawled data repositories.

Troubleshooting performance and scalability

Troubleshooting query performance issues

SharePoint search has an instrumented query pipeline and associated administration reports that help troubleshoot server-based query performance issues. This section will show reports and use them to help understand how to troubleshoot issues on the server. In addition, this section also contains tools and guidance available to assist in addressing client-based (browser) performance issues.

Server-based query issues

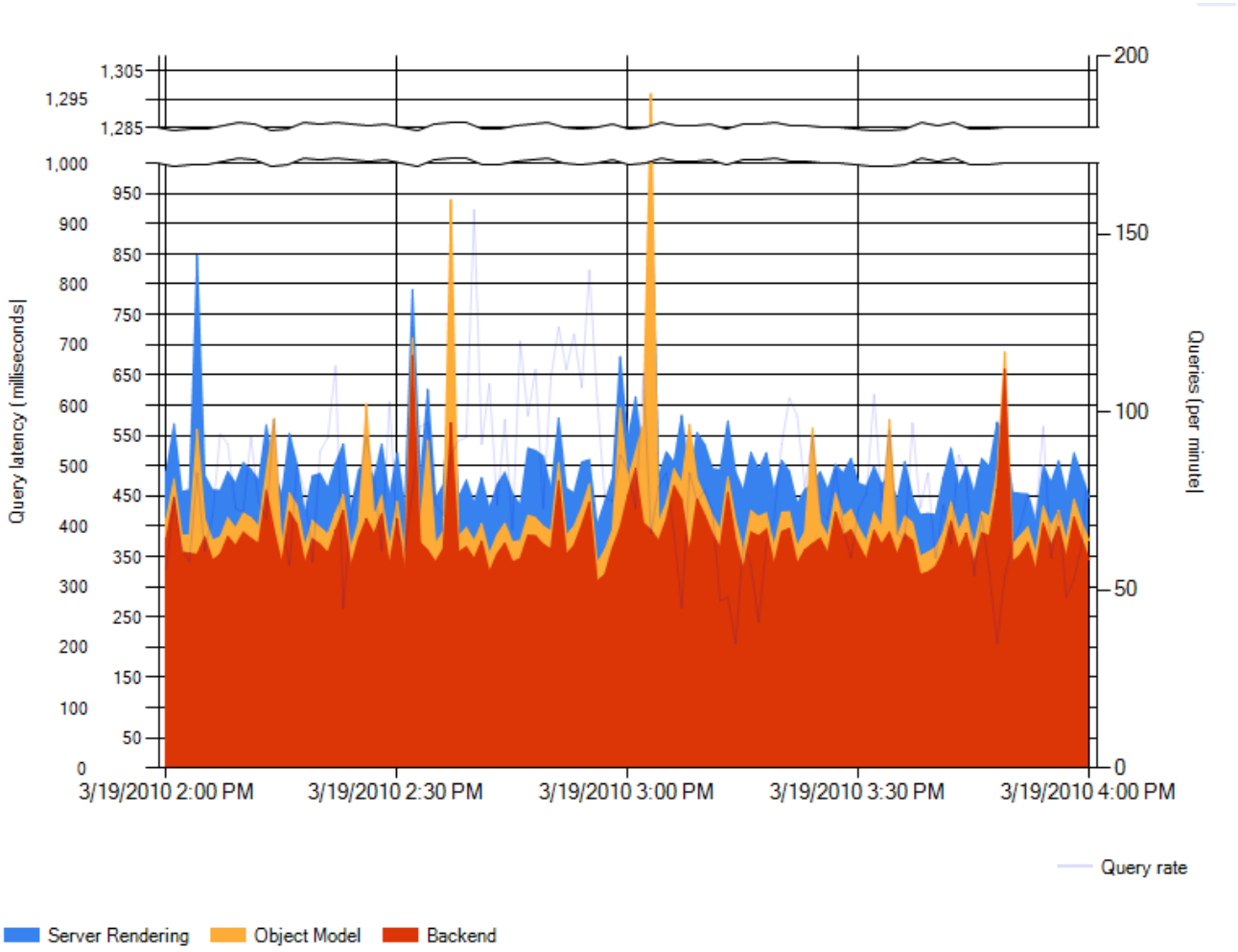
Server-based query performance can be segregated into two levels of issues:

- Search front-end performance issues
- Search back-end performance issues

The following two subsections give the details for troubleshooting each of them. Please note that these are high-level guidelines.

Front-end performance issues

The first step in troubleshooting front-end query performance should be reviewing the **Overall Query Latency** search administration report. Here is an example report:



In this report, front-end performance is represented by the following data series:

- Server Rendering: This value represents, for the given minute, the average time spent per query in the various search Web Parts in the Web front-end.
- Object Model: This value represents, for the given minute, the average time spent in communication between the Web front-end and the search back-end.

Troubleshooting server rendering issues

Server rendering issues can be affected by anything occurring on the Web front-end serving the Search Center results page. In general, you want to understand how much time is being spent in retrieving the various Web Parts in order to find where the extra latency is being added. Enable the Developer Dashboard on the search results page for detailed latency reporting. Common issues that manifest as excess server rendering latency include:

- Platform issues. This includes:
  - Slow Active Directory lookups
  - Slow SQL Server times

  - Slow requests for fetching the user preferences
  - Slow calls to get the user's token from secure token service
- Code-behind issues. This includes modified search results pages (such as results.aspx) that are checked in but not published.

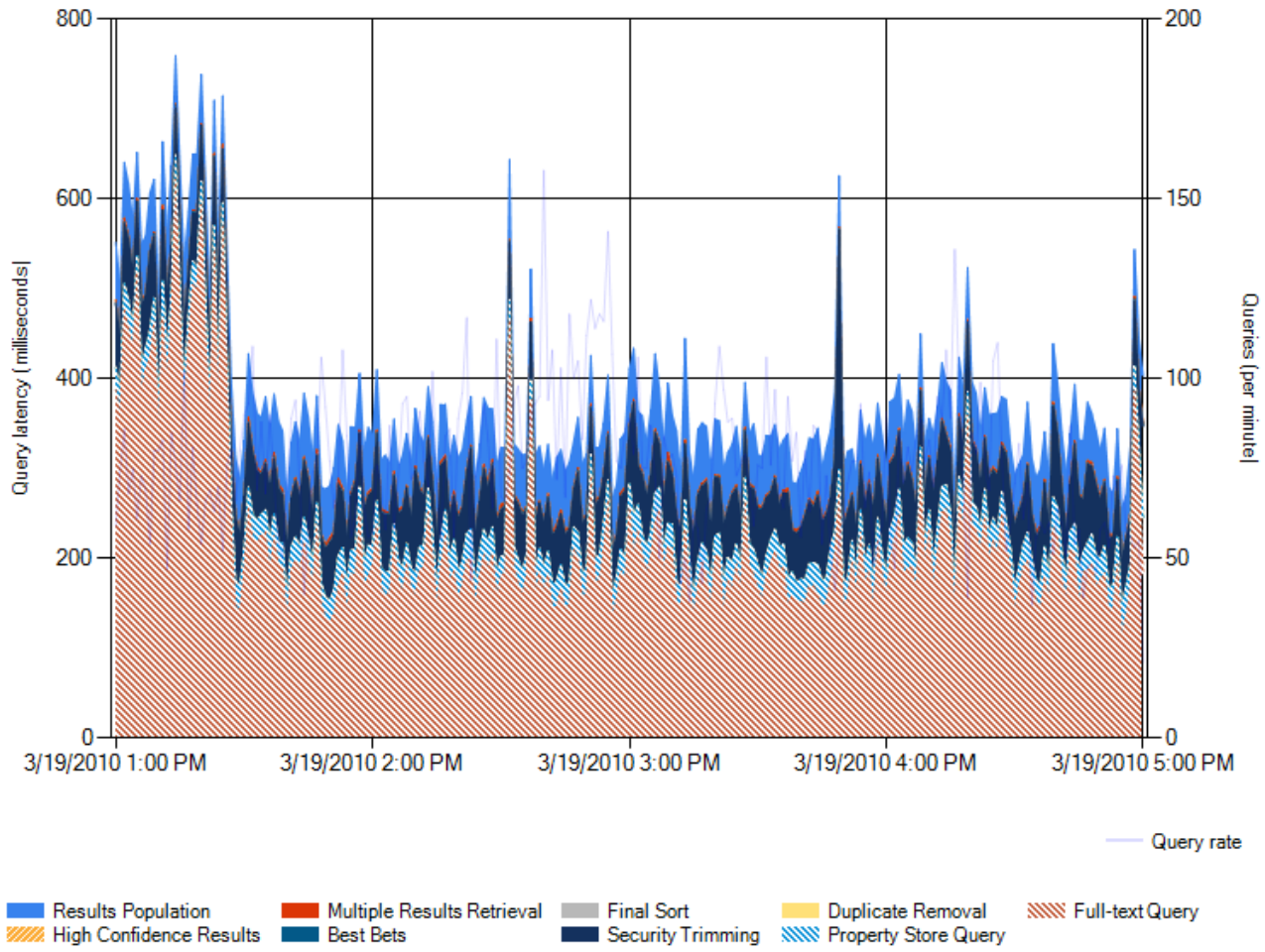
Troubleshooting object model issues

Object model issues can be affected by:

- Issues with the Windows Communication Foundation (WCF) layer:
  - Timeouts and threadabortexception in WCF calls in your deployment.
- Issues with communication between the content and service farms (if configured).

Back-end performance issues

The first step in troubleshooting back-end query performance should be reviewing the **SharePoint Backend Query Latency** search administration report. Here is an example report:



In this report, back-end performance is represented by the following data series (each is average time spent per query, in the given minute), grouped by functional component:

- Query component:
  - Full-text Query: This value shows the average time spent querying the full-text index for results.
- Property database:
  - Multiple Results Retrieval: This value shows the average time spent retrieving document metadata, such as title or author, to appear in the query results.
  - Property Store Query: This value shows the average time spent querying the property database for property-based queries.
- Search Administration database:
  - Best Bets: This value shows the average time spent determining whether there are Best Bets available for the query terms.
  - High Confidence Results: This value shows the average time spent to retrieve high confidence results for queries.
- Query processor:
  - Security Trimming: This value shows the average time spent removing items the user does not have access to.
  - Duplicate Removal: This value shows the average time spent removing duplicates.
  - Results Population: This value shows the average time spent creating the in memory table to be passed back to the object model.

Troubleshooting query component performance issues

Query components are resource intensive, especially when the component is "active"; that is, responding to query requests. Troubleshooting query component performance is one of the more complicated search areas. Here are general areas to consider:

- The most resource intensive query component event is the master merge, where shadow indexes are merged with the master index. This event occurs independently for each query component, although usually simultaneously across all query components. An example of the impact can be seen in the **SharePoint Backend Query Latency** report above, at times prior to 1:30 PM. If this event is impacting end-user query latency, it is possible to define "blackout" periods where a master merge event is avoided unless the percentage of change exceeds the defined limit.
- Sustained high values for the environment mean that you should probably do the following:
  - Examine the index size for each component on the server. Ensure enough RAM on the server exists to allow ~33% of the sum of index sizes to be cached.
  - Examine the query component IO channel on the server. Ensure you are not experiencing an IO bottleneck.

Troubleshooting property database issues

Examine SQL Server health by using concepts in the following article: Storage and SQL Server capacity planning and configuration. If you are executing custom queries, you may need to look at setting a QueryHint parameter on the query, to guide the correct query plan.

Troubleshooting Search Administration database issues

Examine SQL Server health by using concepts in the following article: Storage and SQL Server capacity planning and configuration.

Troubleshooting query processor issues

Troubleshooting query processor issues depends on the area of the query processor that is exhibiting the degraded query latency:

- Security trimming:
  - For Windows claims, examine the Active Directory connection from the server hosting the query processor.
  - For all cases, the cache size used by the QP can be adjusted higher, if there is a correlation between a large number of round trips to SQL Server (determined by SQL profiler). More than 25% of queries should not need a SQL Server call to retrieve security descriptors from the Search Administration database. If they do, adjust the QP cache size.
- Duplicate removal:
  - Look at whether you are crawling the same content in multiple places. Disable duplicate detection in the Search Center.
- Multiple results retrieval:
  - Examine SQL Server health using concepts in the following article: Storage and SQL Server capacity planning and configuration.

Browser-based query issues

Users can be either delighted or exasperated by the speed of search results. Page load time (PLT) is one of the important factors in a user's impression of a user experience. Most of the focus around PLT is on the server-side, namely the time it takes the server to return results. Client-side rendering, however, can compose a significant portion of PLT and is important to consider.

The out-of-box search user experience is designed to provide sub-second responses for total PLT. Out of that time, client rendering typically takes less than 280 milliseconds, depending upon your browser and rendering measurement. This experience delights users with very fast results.

Customizations to the results experience can easily degrade rendering performance. Search administrators and developers must be vigilant in measuring the rendering time after each modification to ensure performance has not regressed significantly. Every addition to the page from a new Web Part to a new cascading style sheets (CSS) style will increase rendering time on the browser and delay results for your users. The amount of delay, however, can vary greatly based on whether certain best practices are followed when customizing the page.

Here are some general guidelines:

- Basic branding and style customizations to the page should not add more than approximately 25 ms to PLT. Measure PLT before and after implementing customizations to observe the change.

- Users typically notice a change (faster or slower) in an increment of 20%. Keep this in mind when making your changes; 20% of the out of the box rendering time is only 50 ms. (Source: Designing and Engineering Time)
- Cascading style sheets (CSS) and JavaScript (JS) are the most common and largest culprits to high rendering performance. If you must have customized CSS and JS, ensure they are minimized to one file each.
- JS can load on-demand after the page renders to provide the user with visible results sooner.
- The more customizations added to the page, the slower it will load. Consider whether the added functionality and style is worth the extra delay on results for your users.

In addition to these guidelines, there is a wealth of information and tips on the Internet about how to reduce PLT and the effect of slow pages on a user experience. Read up on the literature and delight your users with fast pages!

Troubleshooting crawl performance issues

Search Server 2010 Express may experience bottlenecks in the crawl sub-system as the system moves through index acquisition, maintenance and deletion phases. To effectively troubleshoot crawl performance issue, you should use the Search Health Monitoring Reports in conjunction with the common bottlenecks and their causes section to isolate crawl issues.

Troubleshooting during the index acquisition phase

The first place to identify crawl issues is the Crawl Rate Per Content Source health report. In general, the full crawl rate should be greater 35 docs/sec for all other types of content sources.

Once the content source with suboptimal crawl rate is identified, the following steps are recommended:

1. Pause all other crawls except the content source under investigation. Did the crawl rate improve beyond the specified 15/35 docs/sec goal?
2. If the above does not help, then ensure that the repository being crawled is responsive enough and not the cause for slow crawl. See Crawl bottleneck on repository earlier in this document.
3. If the repository is not the bottleneck, the next step is to identify the bottleneck in the indexer or database servers and optimize around them. Guidance can be found in the Crawl IOPS Bottleneck and Crawl CPU Thread Bottleneck sections.

Troubleshooting during Content Maintenance Phase

The primary goal during the content maintenance phase is to keep the search index as fresh as possible. Two of the key indicators for this are index freshness and incremental crawl speed.

1. **Index freshness:** Are the crawls finishing in their budgeted time and in accordance with the IT guidelines for index freshness?
2. **Incremental crawl speed:** If the index freshness goal is not met, then the first step is to investigate if whether the incremental crawl speeds are 25 docs/sec for the content sources. If the incremental crawl speeds are suboptimal, a bottleneck analysis should be performed on the crawled repository and the crawl subsystem as described above.

Common bottlenecks and their causes

During performance testing, several different common bottlenecks were revealed. A bottleneck is a condition in which the capacity of a particular constituent of a farm is reached. This causes a plateau or decrease in farm throughput.

The following table lists some common bottlenecks and describes their causes and possible resolutions.

Bottleneck	Symptom (perf counter)	Resolution
Database RAM	Property database, Search administration database exhibit: • SQL Server Buffer Manager/ Page Life Expectancy < 300(ms) *(should be > 1000 (ms))* • SQL Server Buffer Manager/ Buffer Cache Hit Ratio < 96% *(should be > 98%)*	• Add more memory to the database server, if it is separate. • Defragment the property database, if the weekly defrag rule has been disabled. • Ensure you are using SQL Server 2008 Enterprise edition, to enable page compression.
Database server IOPS	A property database or crawl database exhibits: • Average disc sec/read and Average disc sec/write ~50 ms or > 50 ms	• Increase the dedicated number of IOPS for the database: ○ Use different storage arrays ○ Optimize your storage configuration; for example, by adding spindles (disk drives) to the storage array. • Run SPHA property database defragment rule, if it has been disabled. • Run SPHA crawl database defragment rule. • Ensure you are using SQL Server 2008 Enterprise edition, to enable page compression.
Query component IOPS	The logical disk used for a query component's index exhibits: • Average disc sec/read and Average disc sec/write ~30 ms or > 30 ms for a sustained period of time (that is, most of the day, not just during an	• Increase the dedicated number of IOPS for the drive used for the query component's index: ○ Use different storage arrays for different components. ○ Optimize your storage configuration; for example, by adding spindles (disk drives) to the storage array.

	index merge).	