Design Search Architectures for Microsoft SharePoint Server 2010

This poster will help you go through the initial design steps to determine a basic design for a Microsoft® SharePoint® Server 2010 search architecture. Each successive step will help you refine the initial design by identifying key design drivers, starting with business requirements and metrics. You can use the outputs of each step to inform the next set of questions. After going through each step in this poster, you will be able to map business requirements and key performance metrics to a baseline search architecture.

Search model 4 of 4

1 Identify corpus volume and key performance metrics

The number of items (sites, lists, items in document libraries, etc.) in the organization plays a key role in determining architectural requirements for search.

The following table describes how the number of items you plan to crawl affects design decisions. Use this information to determine a starting-point architecture. For examples of starting-point architectures, see Poster 3 in this 4-part series.

The starting point architecture you select in this step may change depending on your requirements in the subsequent steps.

Number of Items	Starting point architecture
0-1 million	Limited deployment
1-10 million	Small farm topology
10-20 million	Medium shared farm topology
20-40 million	Medium dedicated farm topology
40-100 million	Large dedicated farm topology

You now have to determine the importance and relative priority of performance requirements for the environment.

The following table lists the different variables that compose the "big picture" of overall performance. The relative importance of these variables in the environment is generally driven by a service level agreement (SLA). Similarly, the SLA affects certain design considerations.

For example, if you know that query results must be returned in <1 second on average, low query latency is a key requirement. If you also expect a large average volume of concurrent queries, these two factors together suggest the need for multiple query servers and possibly several index partitions. If these factors are more important than crawl speed (for example, if the content to be crawled is fairly small in volume), you should allocate more resources to the query role than the crawl server role.

This metric	Is affected by these factors
Full crawl time and result freshness	• Number of data sources • Data source response time • Size and type of files • Network bandwidth • Query load while crawling
Time required for results to be returned	• Number of concurrent user queries • Number of applications using Search
Availability of query functionality	• Hardware availability
Availability of content crawling and indexing functionality	• Hardware availability

Example – Company A Step 1

In this example, Company A's corpus contains 10-20 million items. Based on this corpus size, the best starting point architecture is the medium shared search topology from Poster 3.

Company A's business requirements include a very low average query latency, with search results returned in less than one second on average.

Content to be crawled spans multiple content locations, some of which may be across low-bandwidth WAN connections.

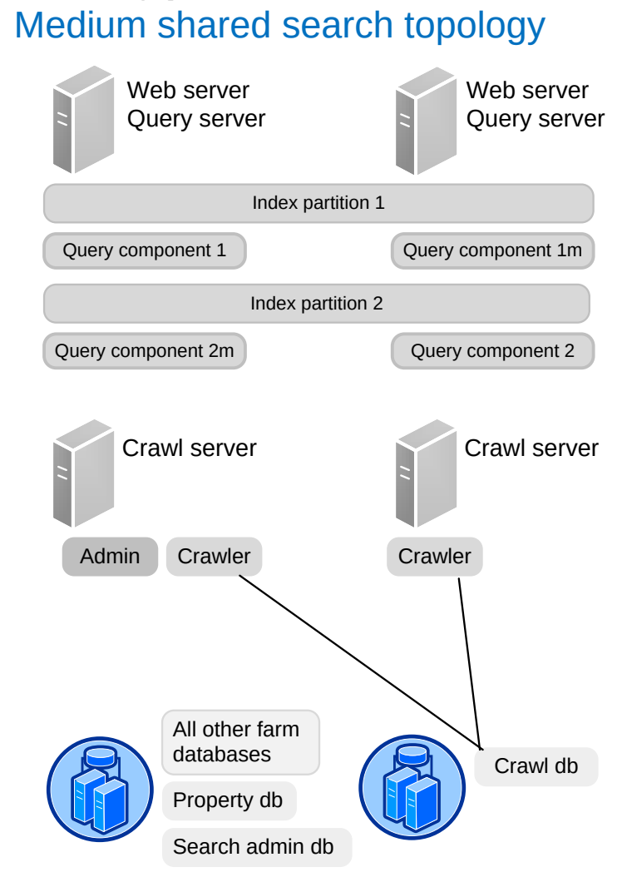
A high peak query load is not expected, but content freshness is important. This means that crawl speeds must be fast, and it is important that overall crawling is not delayed if some content locations are slow to respond.

Requirements:

- Low query latency
- Fast crawl speed
- Concurrent crawls of different content sources

Number of items:
<20 million

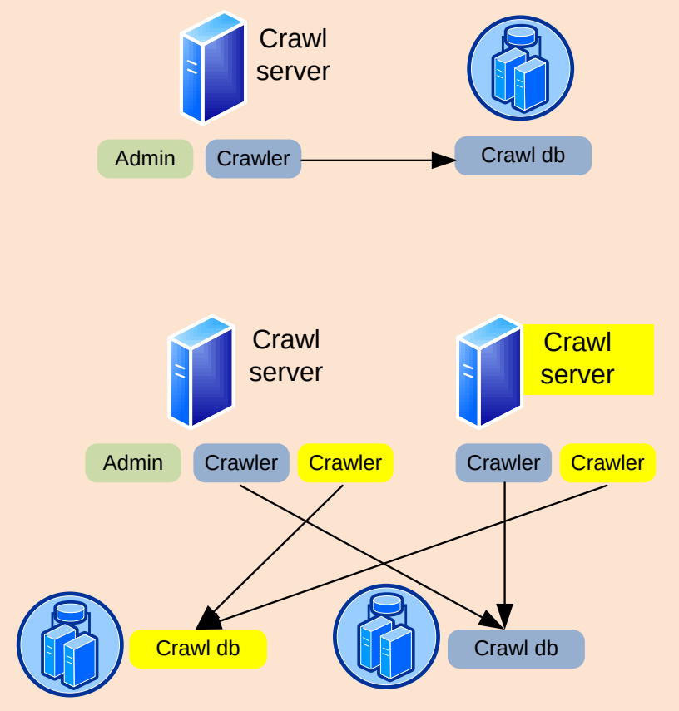
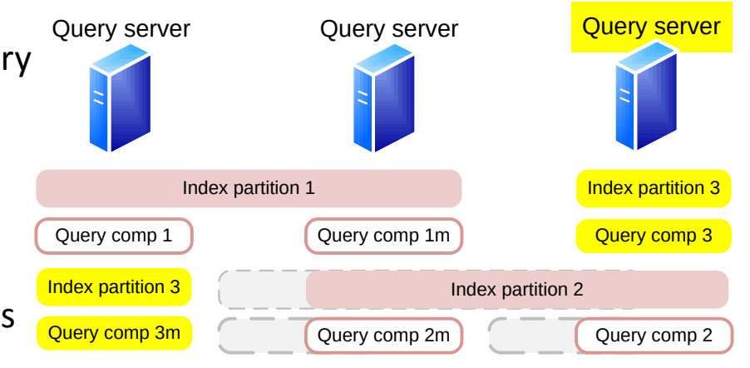
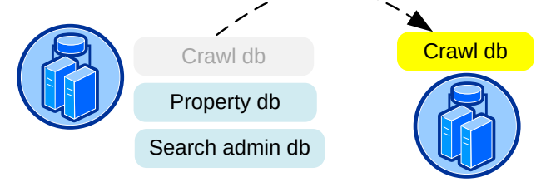
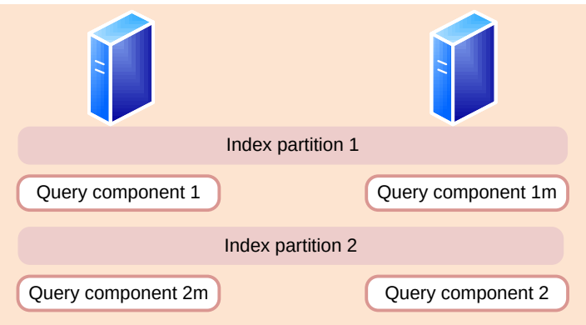
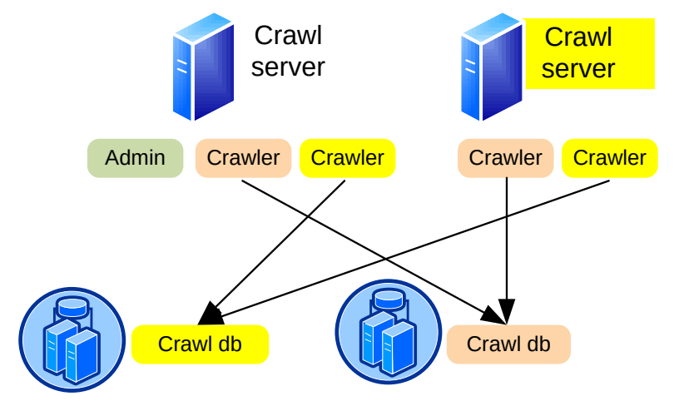
Starting point architecture:
Medium shared search topology



2 Map key metrics to logical topology and search components

In this step, map key performance metrics and business requirements to specific logical topology choices.

Use this table to decide how to distribute logical components to support the performance metrics and other requirements you identified in Step 1.

To satisfy this metric	Take these actions	
Full crawl time and result freshness	Add crawl servers, crawlers, and crawl databases. Each crawl database can contain content from independent sources. Each crawl database can have several crawlers associated with it, and those crawlers can be distributed among multiple crawl servers. The more crawlers you use, the more parallel crawl processes can be performed. If you have several content locations to crawl, multiple crawlers and associated crawl databases enable you to crawl them concurrently. Because each crawler runs parallel to the others, overall crawl time is reduced as you add crawlers.	 Crawl server Admin Crawler Crawl db Crawl server Crawl server Admin Crawler Crawler Crawler Crawler Crawl db Crawl db
Time required for results to be returned	If query latency or low query throughput is caused by high peak query load, deploy new query servers and multiple query components for existing index partitions across query servers. Each index partition can contain up to ~10 million items. If you have more than 10 million items per index partition, add index partitions and distribute their query components across multiple query servers. If query latency or low query throughput is caused by database load, isolate the property database from crawl databases by moving it to a separate database server. If you have over 25 million items in your corpus, or a large volume of metadata, you may need to add another property database.	 Query server Query server Query server Index partition 1 Index partition 3 Query comp 1 Query comp 1m Query comp 3 Index partition 3 Index partition 2 Query comp 3m Query comp 2m Query comp 2
Availability of query functionality	Deploy redundant query servers, redundant index partitions and query components, and use clustered or mirrored database servers to host crawl and property databases.	 Index partition 1 Query component 1 Query component 1m Index partition 2 Query component 2m Query component 2
Availability of content crawling and indexing functionality	Use multiple crawlers on redundant crawl servers, and add crawl databases. Crawlers associated with a given crawl database can be distributed across crawl servers for availability and load distribution.	 Crawl server Crawl server Admin Crawler Crawler Crawler Crawler Crawl db Crawl db

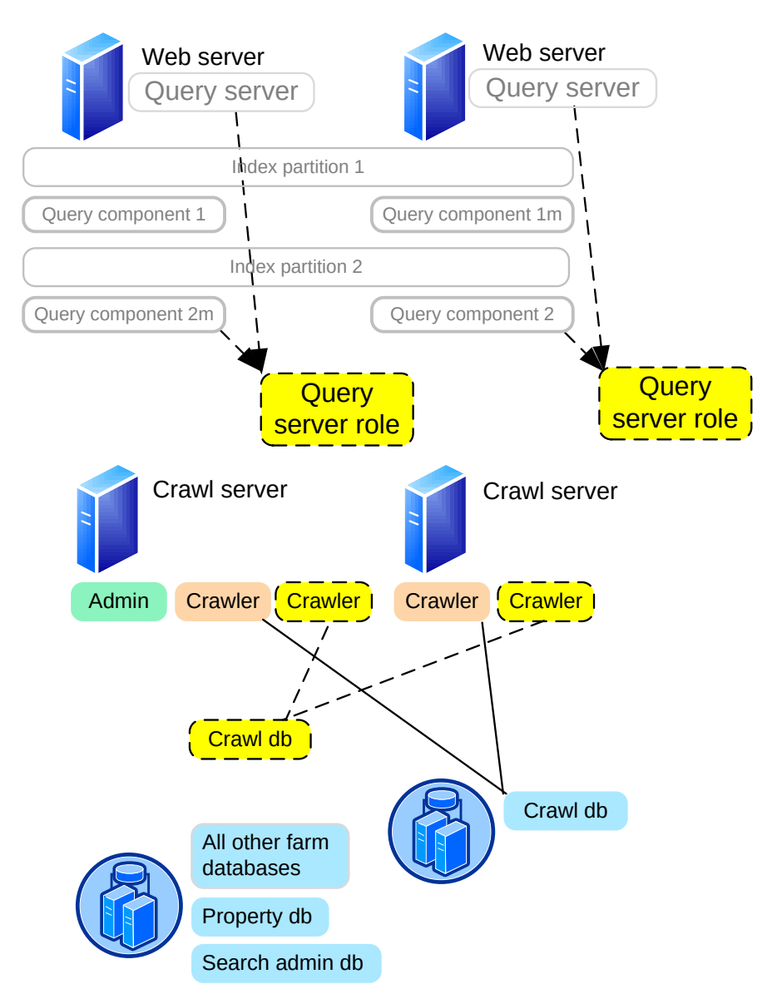
Example – Company A Step 2

Company A's requirements from Step 1 were primarily low query latency and fast crawl speed.

The number of items in the corpus (<20 million) requires a minimum of two index partitions, which can effectively contain up to 10 million items each. If substantial growth of the corpus is expected, more index partitions can be added to anticipate a higher volume of items.

To help ensure low query latency, the query server role should be separated from the Web server role. Multiple index partitions and query components are needed.

Because Company A requires high crawl speed and daily crawls to maximize freshness of query results, multiple crawl servers and crawlers are needed. Because some of the content locations may be slow to respond, additional crawlers and a new crawl database should be added to increase parallelization of the crawl process. This will help to ensure that crawling of all content locations does not take more than 24 hours.



3 Map logical topology to a physical architecture design

In this step, you will see how logical topology requirements map to hardware and physical architecture design considerations. Use the information in the table below to determine the physical servers and topology you will need to support the logical topology requirements you identified in Step 2.

Hardware component	Logical component	# of items (in millions):					Physical topology considerations
		<10	10-40	40-60	60-80	80-100	
Query server	Query server role	Shared with crawl server role, or 1-2 independent query servers	2-4	5-6	7-8	9-10	The number of query servers is dependent on the volume of queries, the number of items in the corpus, and redundancy and availability requirements. The query server role can run on the same server with any other SharePoint services or by itself. If you expect a large volume of query traffic, and require low query latency, consider deploying at least one dedicated query server.
	Index partition (2 per query server)						Each index partition contains a discrete portion of the corpus, and can contain up to 10 million items. Each index partition can be "mirrored" using query components (see below). We recommend that you deploy one query server for every two index partitions you add.
	Query component	1-4	4-8	10-12	14-16	18-20	Query components are mirror copies of a given index partition. Query components associated with the same index partition can be distributed among several query servers for redundancy and to improve query performance. Generally, two query components for a given index partition, each hosted on a different query server, are sufficient to fulfill performance and redundancy requirements.
Crawl server	Crawl server role	Shared with query role, or 1 independent crawl server	1-2	2	3	3-4	The crawl server role can run on the same server with any other SharePoint services or by itself. If you expect to crawl a large volume of content, plan to crawl content across a variety of sources, or require that crawling takes place while queries are being performed, consider deploying at least one dedicated crawl server.
	Crawler (2 per crawl server)	1-2	1-4	4	6	6-8	You can have as many crawlers on a given crawl server as resources permit, but we recommend two per crawl server. If you have a variety of content sources, you can add crawlers and crawl databases and dedicate them to specific sources. Each crawler on a given crawl server should be associated with a separate crawl database. For example, if you have two crawl servers and four crawlers, you should have two crawl databases. See the example diagram below for details.
Database server	Crawl database	1	1-2	2	3	3-4	The crawl database contains crawled content, and should be maintained on a separate hard disk from the property database as a best practice to prevent I/O contention. If the crawl window overlaps with times when users are querying, or several crawlers are connected to a crawl database, consider deploying the crawl database to a separate database server. You can also have multiple crawl databases with different crawlers connected to them.
	Property database	1	1-2	2	3	3-4	The property database contains metadata for all crawled content. You may need more than one property database per 25 million items if there is a large amount of metadata associated with crawled content.
	Search Admin database	1	1	1	1	1	Only one Search Administration database is required per farm.

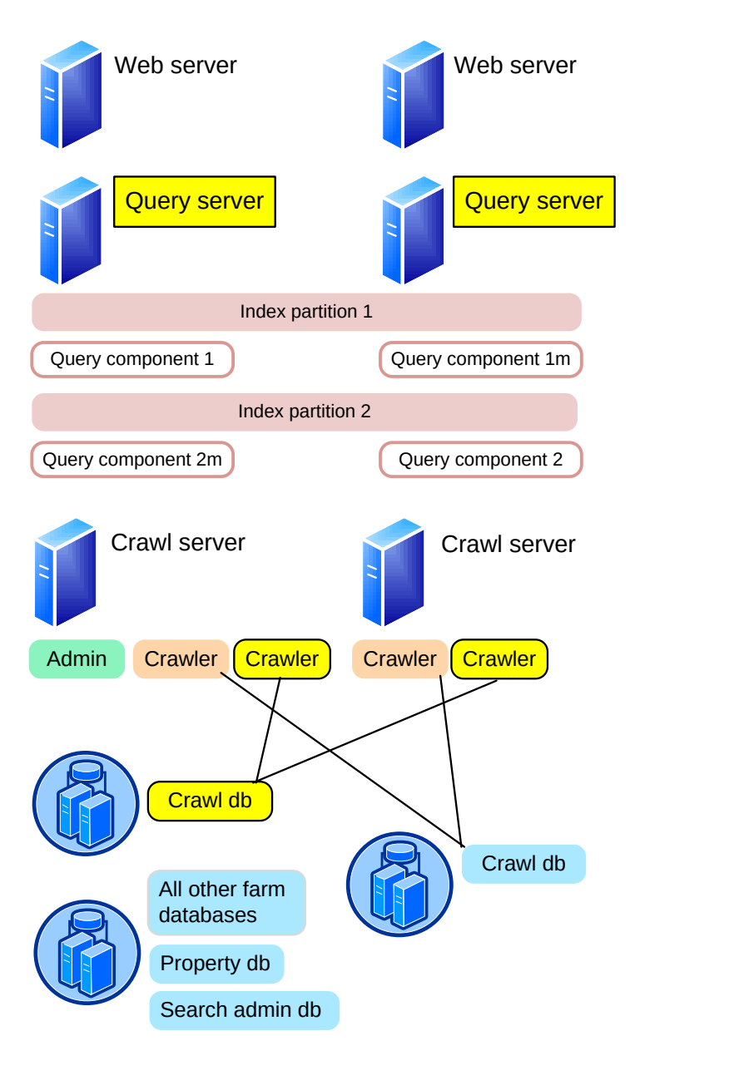
Example – Company A Step 3

In this step, Company A's starting-point physical architecture is revised to support the logical topology requirements that were identified in step 2.

To ensure adequate query throughput, the query role has been deployed to twp separate query servers, isolating query functionality from Web server demand and maintaining redundancy of the query server role. This will help to ensure low query latency, and make it easier to scale out this role to meet future needs without affecting Web server performance.

Two crawl servers are required for redundancy and to help ensure that crawl speed is fast enough to maintain content freshness. A crawler is added to each of the two crawl servers to accommodate multiple content locations, and a new crawl database is added on its own database server to service the new crawlers.

In this particular case, the volume of metadata associated with crawled content is estimated to be relatively small, so one property database should be sufficient. However, if the volume of metadata increases over time, or if the volume of the corpus approaches ~25 million items, another property database on an additional database server will be needed to maintain adequate query throughput.

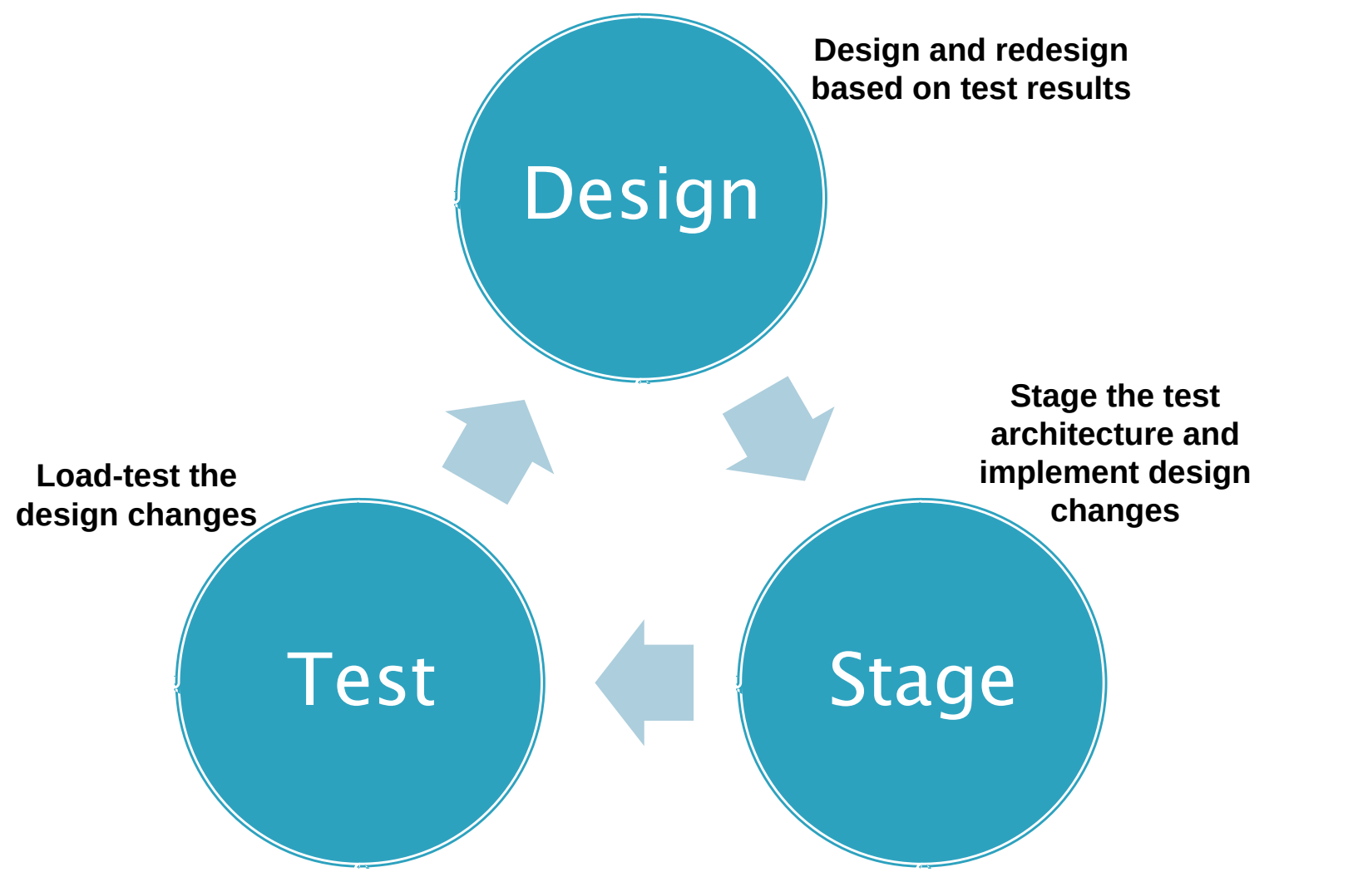


As shown in the example above, crawl databases should always run on separate database servers from the property database to prevent resource contention from queries when crawling is taking place.

In environments where crawl freshness is a priority, the crawl database should generally be hosted on its own database server.

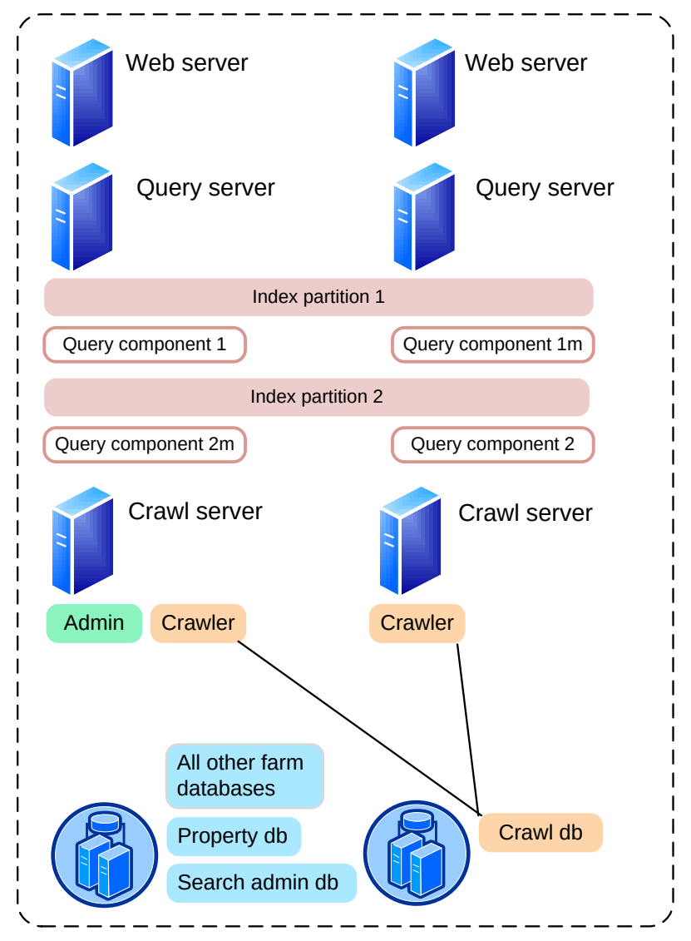
4 Stage, test, and iterate

Now that you have an initial search architecture design, deploy to a staging environment, and then test to identify weak points in the design. You can then change the design to resolve issues before you deploy to a production environment.


Design
Design and redesign based on test results
Stage the test architecture and implement design changes
Stage
Test
Load-test the design changes

Stage

Using the decisions that you made by following the steps in this poster to drive the initial design, deploy the initial farm design to a staging environment. You should build the staging environment to exactly correspond to the production design to ensure that test results accurately reflect the behavior of the production environment.

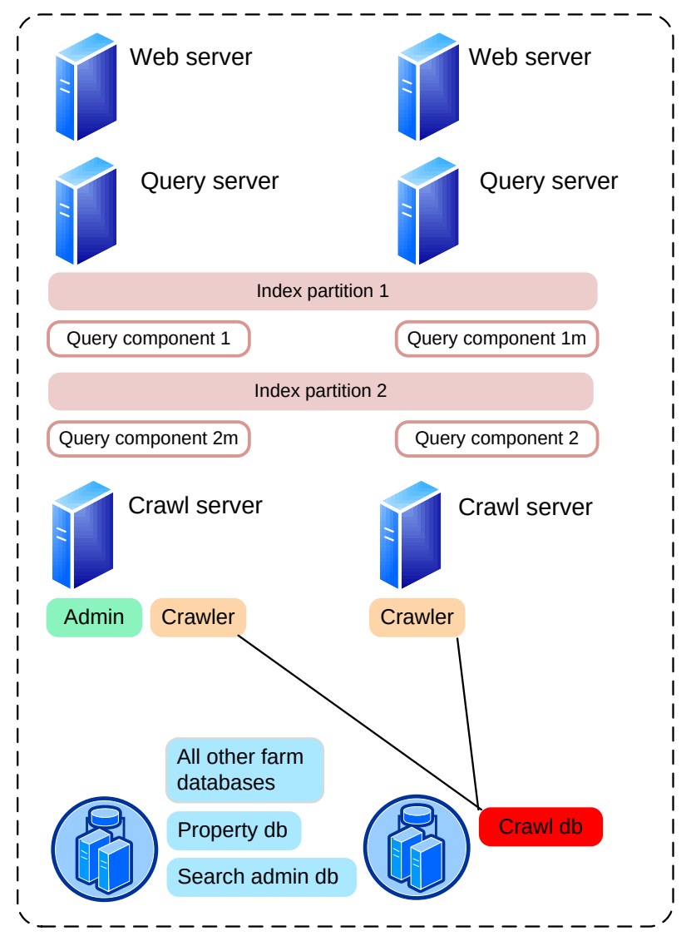


Test

Conduct load testing against the staged deployment. Load testing will reveal any weak points in the design.

In this example, test results reveal that when an expected volume of queries and other farm activity occur at the same time that crawling occurs, query latency increases to an unacceptable level because of resource contention on the database server hosting the crawl database and other farm databases.

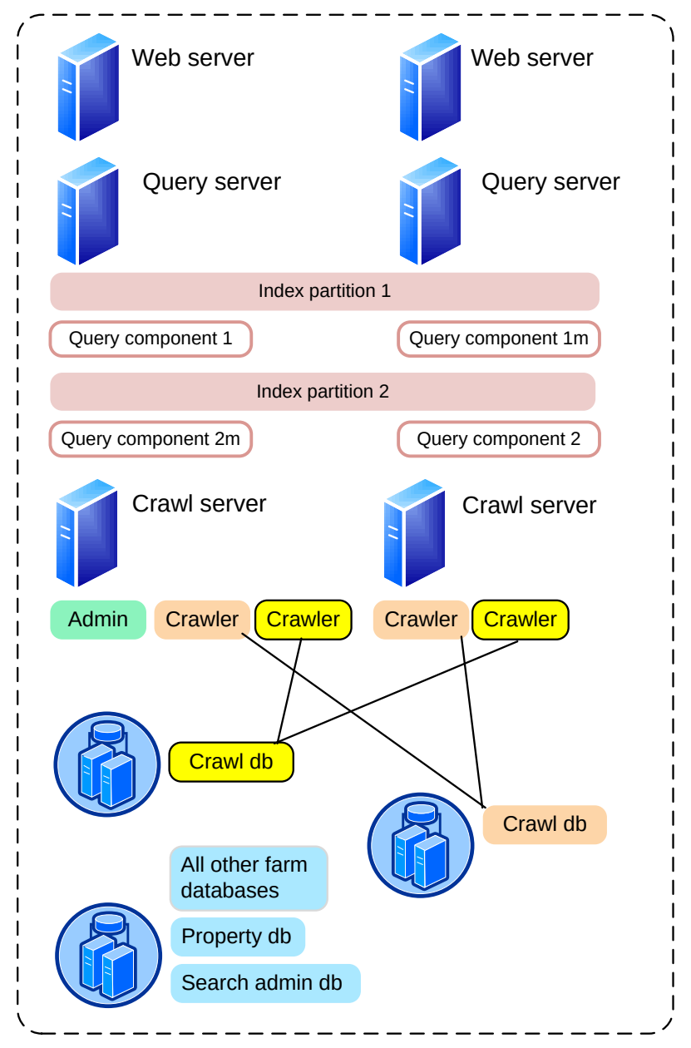
This bottleneck is revealed through observation of unacceptably high disk I/O and large disk queue lengths on the database server that is hosting the crawl database.



Iterate

To resolve the problem that testing revealed, a third database server is added, and a second crawl database is added to the new server. Two new crawlers are added, one on each crawl server, to crawl content for the new crawl database.

After changes have been made to resolve the problem revealed in testing, test again to see whether the changes actually resolved the problem and also to identify any other problems that may have been masked by issues that have now been resolved.



This document supports a preliminary release of a software program that bears the project code name Microsoft® SharePoint® Server 2010 Beta.

To send feedback about this documentation, please write to us at ITSPdocs@microsoft.com.