

4.8 - Maintain an Azure AI Search solution

- [Overview](#)
- [Learning objectives](#)
- [Introduction](#)
- [Manage security of an Azure AI Search solution](#)
 - [Overview of security approaches](#)
 - [Data encryption](#)
 - [Secure inbound traffic](#)
 - [Authenticate requests to your search solution](#)
 - [Secure outbound traffic](#)
 - [Secure data at the document-level](#)
- [Optimize performance of an Azure AI Search solution](#)
 - [Measure your current search performance](#)
 - [Check if your search service is throttled](#)
 - [Check the performance of individual queries](#)
 - [Optimize your index size and schema](#)
 - [Improve the performance of your queries](#)
 - [Use the best service tier for your search needs](#)
- [Manage costs of an Azure AI Search solution](#)
 - [Estimate your search solutions baseline costs](#)
 - [Understand the billing model](#)
 - [Tips to reduce the cost of your search solution](#)
 - [Manage search service costs using budgets and alerts](#)
- [Improve reliability of an Azure AI Search solution](#)
 - [Make your search solution highly available](#)
 - [Distribute your search solution globally](#)
 - [Back up options for your search indexes](#)
- [Monitor an Azure AI Search solution](#)
 - [Monitor Azure AI Search in Azure Monitor](#)
 - [Use metrics to see diagnostic data visually](#)
 - [Write Kusto queries against your search solutions logs](#)
 - [Create alerts to be notified about common search solution issues](#)
- [Debug search issues using the Azure portal](#)
 - [Explore how to use the Debug Session tool in Azure AI Search](#)
 - [Debug a skillset with Debug Sessions](#)
 - [Create a Debug Session](#)
 - [Explore and edit a skill](#)
 - [Validate the field mappings](#)
- [Exercise - Debug search issues](#)
 - [Create your search solution](#)
 - [Import sample data and configure resources](#)
 - [Use a debug session to resolve warnings on your indexer](#)

- [Resolve the warning on the indexer](#)
 - [Clean-up](#)
 - [Knowledge Check](#)
 - [Summary](#)
-

Overview

Maintain the performance, cost, and reliability of your Azure AI Search solutions.

Learning objectives

In this module, you'll learn how to:

- Use Language Studio to enrich Azure AI Search indexes
 - Enrich an AI Search index with custom classes
-

Introduction

Running a successful Azure AI Search solution requires you to understand how to maintain its **two primary workloads of indexing and querying**. It's essential that the search solution is as **cost effective** as possible.

In this module, you'll learn how to:

- Manage the security of your search service and source data.
- Optimize the performance your indexes and manage costs
- Improve reliability, monitor the performance and run queries against Log Analytics.
- Debug indexer related errors and warnings.

Note: This module assumes you already know how to create and use an Azure AI Search solution that includes built-in skills. If not, complete the [Create an Azure AI Search solution](#) module first.

Manage security of an Azure AI Search solution

Organizations need to be able to trust the security of their search solutions. Azure AI Search gives you control over how to secure the data you search.

Here, you'll explore how to secure your search solution. You'll focus on where data is **encrypted** and how to **secure the inbound and outbound data flows**. Finally, you'll see how to **restrict access** to search results for specific users or groups.

Overview of security approaches

AI Search security builds on Azure's existing network security features. When you think about securing your search solution, you can focus on three areas:

- Inbound search requests made by users to your search solution
- Outbound requests from your search solution to other servers to index documents
- Restricting access at the document level per user search request

Data encryption

The Azure AI Search service, like all Azure services, encrypts the data it stores at rest with service-managed keys. This encryption includes indexes, data sources, synonym maps, skillsets, and even the indexer definitions.

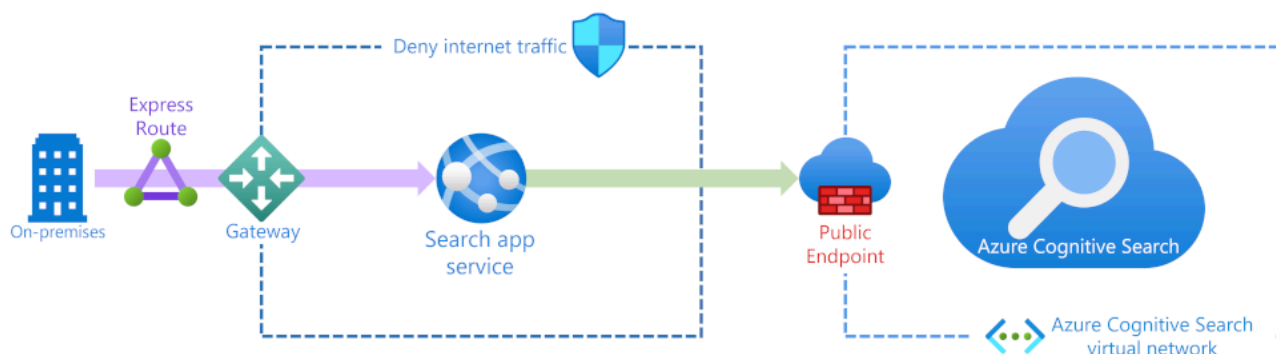
Data in transit is encrypted using the standard HTTPS TLS 1.3 encryption over port 443.

If you'd like to use your own encryption keys, the use of Azure Key Vault is supported. A benefit of using your own customer-managed keys is that **double encryption** will be enabled on all objects you use your custom keys on.

Note: For detailed steps on how to use customer-managed keys for encryption, see [Configure customer-managed keys for data encryption in Azure AI Search](#)

Secure inbound traffic

If your search solution can be accessed externally from the internet or apps, you can reduce the attack surface. Azure AI Search lets you restrict access to the public endpoint for free using a firewall to allow access from specific IP addresses.



If your search service is only going to be used by on-premises resources, you can harden security with an ExpressRoute circuit, Azure Gateway, and an App service.

There's also the option to change the public endpoint to use an Azure private link. You'll also need to set up an Azure virtual network and other resources.

Using a private endpoint is the most secure solution, although it does come with the added cost of using those services that need to be hosted on the Azure platform.

Note: For more information about private endpoints, see [Create a Private Endpoint for a secure connection to Azure AI Search](#).

Authenticate requests to your search solution

With the infrastructure in place to reduce the attack surface of your search solution, your focus can change to how to authenticate search requests from your users and apps.

The default option when you create your ACS is key-based authentication. There are two different kinds of keys:

- **Admin keys** - grant write permissions and the right to query system information (*maximum of 2 admin keys can be created per search service*)
- **Query keys** - grant read permissions and are used by your users or apps to query indexes (*maximum of 50 query keys can be created per search service*)

Role-based access control (RBAC) is provided by the Azure platform as a global system to control access to resources. You can use RBAC in Azure AI Search in the following ways:

- Roles can be granted access to administer the service
- Define roles with access to create, load, and query indexes

The built-in roles you can assign to manage the Azure AI Search service are:

- **Owner** - Full access to all search resources
- **Contributor** - Same as above, but without the ability to assign roles or change authorizations
- **Reader** - View partial service information

If you need a role that can also manage the data plane for example search indexes or data sources, use one of these roles:

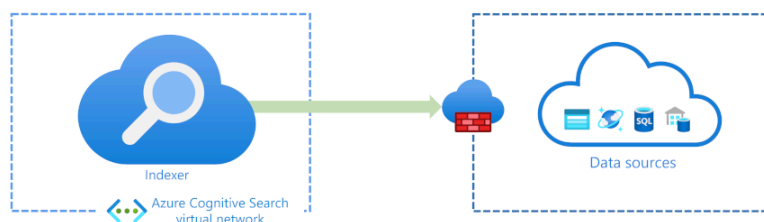
- **Search Service Contributor** - A role for your search service administrators (the same access as the Contributor role above) and the content (indexes, indexers, data sources, and skillsets)
- **Search Index Data Contributor** - A role for developers or index owners who will import, refresh, or query the documents collection of an index
- **Search Index Data Reader** - Read-only access role for apps and users who only need to run queries

Note: For more information about authenticating with RBAC, see [Use Azure role-based access controls \(Azure RBAC\) in Azure AI Search](#).

Secure outbound traffic

Typically your outbound traffic indexes source data or enriches it using Artificial Intelligence (AI). The outbound connections support using key-based authentication, database logins, or Microsoft Entra logins if you can use Microsoft Entra ID.

If your data sources are hosted on the Azure platform, you can also secure connections using a system or user-assigned managed identity.



Azure services can restrict access to them using a firewall. Your firewall can be configured to only allow the IP address of your Azure AI Search service. If you're enriching your indexes with AI, you'll also need to allow all the IP addresses in the **AzureCognitiveSearch** service tag.

You can choose to secure your source data behind a shared private link that your indexers use.

Note: A shared private link requires either a Basic tier for text-based indexing or a Standard 2 (S2) tier for skills-based indexing. For pricing details, see [Azure Private Link pricing](#).

Secure data at the document-level

You can configure Azure AI Search to restrict the documents someone can search, for example, restrict searching contractual PDFs to people in your legal department.

Controlling who has access at the document level requires you to **update each document in your search index**.

You need to add a new security field to every document that contains the user or group IDs that can access it. The security field needs to be filterable so that you can filter search results on the field.

With this field in place and populated with the allowed user or groups, you can restrict results by adding the `search.in` filter to all your search queries. If you're using HTTP POST requests, the body should look like this:

```
{
  "filter": "security_field/any(g:search.in(g, 'user_id1, group_id1, group_id2'))"
}
```

This would filter the returned search results on the user ID and groups that this user belongs to. If your application can use Microsoft Entra ID, it's possible to use the user's identity and group memberships from there.

Note: For a step-by-step guide on how to use Microsoft Entra ID, see [Security filters for trimming Azure AI Search results using Active Directory identities](#)

Optimize performance of an Azure AI Search solution

Your search solutions performance can be affected by the size and complexity of your indexes. You also need to know how to write efficient queries to search it and choose the right service tier.

Here, you'll explore all these dimensions and see steps you can take to improve the performance of your search solution.

Measure your current search performance

You can't optimize when you don't know how well your search service performs. Create a baseline performance benchmark so you can validate the improvements you make, but you can also check for any degradation in performance over time.

To start with, enable diagnostic logging using Log Analytics:

- In the Azure portal, select **Diagnostic settings**.

- Select **+ Add diagnostic settings**.

Dashboard > acs-cognitive-search-service

acs-cognitive-search-service | Diagnostic settings

Search service

Search (Ctrl+/) « Refresh Feedback

Diagnostic settings are used to configure streaming export of platform logs and metrics for a resource to the destination of your choice. You may create up to five different diagnostic settings to send different logs and metrics to independent destinations. [Learn more about diagnostic settings](#)

Name	Storage account	Event hub	Log Analytics workspace	Partner solution	Edit setting
No diagnostic settings defined					
+ Add diagnostic setting					

Click 'Add Diagnostic setting' above to configure the collection of the following data:

- OperationLogs
- AllMetrics

Monitoring

- Alerts
- Metrics
- Diagnostic settings**
- Logs

Automation

- Tasks (preview)
- Export template

- Give your diagnostic setting a name.
- Select **allLogs** and **AllMetrics**.
- Select **Send to Log Analytics workspace**.
- Choose, or create, your Log Analytics workspace.

Home > search-service-acscognitive-search-service | Overview > acs-cognitive-search-service | Diagnostic settings >

Diagnostic setting

Save Discard Delete Feedback

A diagnostic setting specifies a list of categories of platform logs and/or metrics that you want to collect from a resource, and one or more destinations that you would stream them to. Normal usage charges for the destination will occur. [Learn more about the different log categories and contents of those logs](#)

Diagnostic setting name *

Logs

Category groups ⓘ

☐ audit ☒ allLogs

Categories

☒ Operation Logs

Metrics

☒ AllMetrics

Destination details

☒ Send to Log Analytics workspace

Subscription
AI Subscription

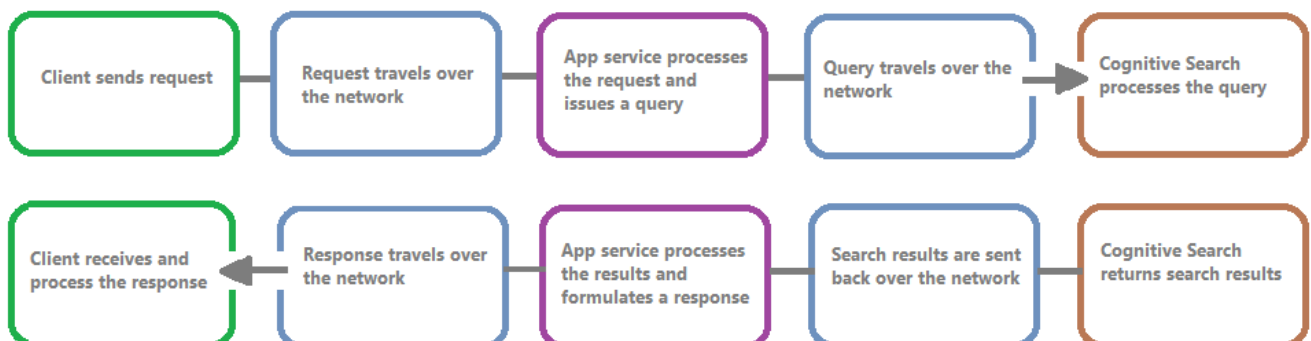
Log Analytics workspace
acs-performance-log-analytics-workspace (westus2)

☐ Archive to a storage account

☐ Stream to an event hub

☐ Send to partner solution

It's important to capture this diagnostic information at the search service level. As there are several places where your end-users or apps can see performance issues.



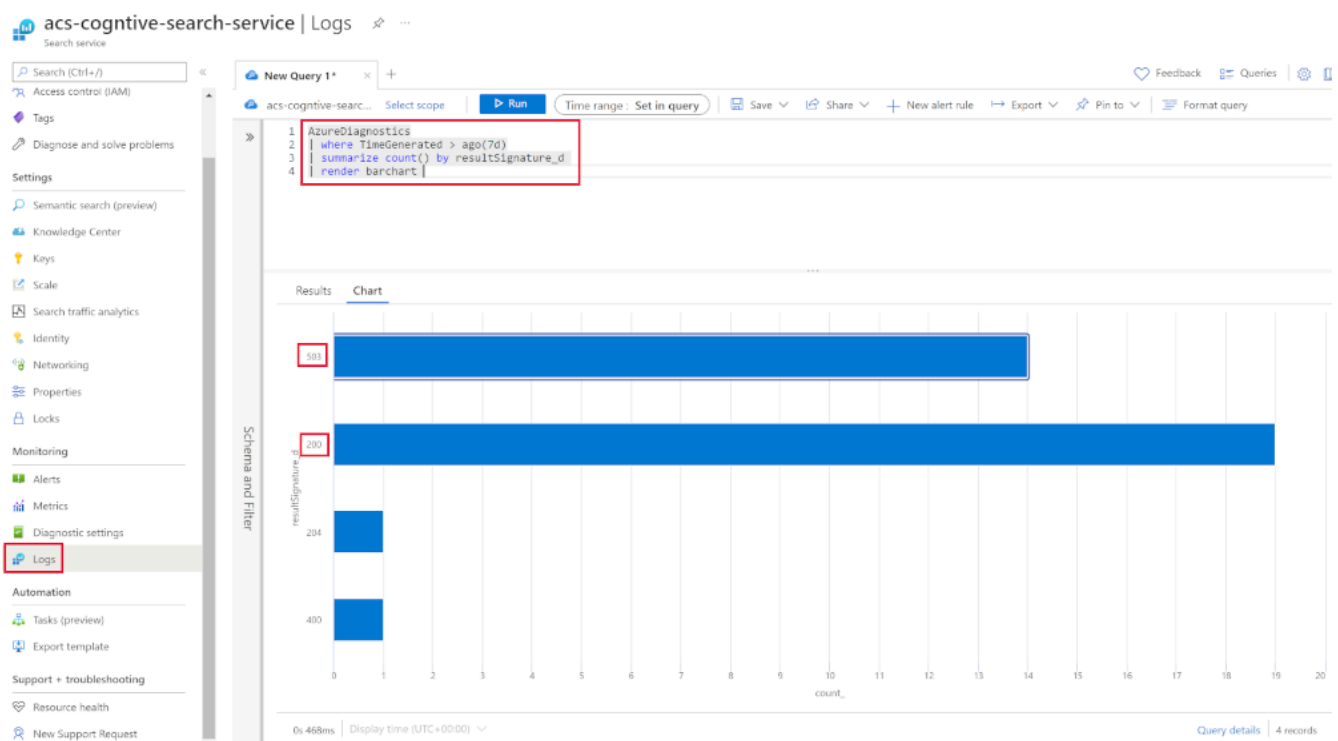
If you can prove that your search service is performing well, you can eliminate it from the possible factors if you're having performance issues.

Check if your search service is throttled

Azure AI Search searches and indexes can be throttled. If your users or apps are having their searches throttled, it's captured in Log Analytics with a 503 HTTP response. If your indexes are being throttled, they'll show up as 207 HTTP responses.

This query you can run against your search service logs shows you if your search service is being throttled. In the Azure portal, under **Monitoring**, select **Logs**. In the New Query 1 tab, you would use this Kusto query:

```
AzureDiagnostics | where TimeGenerated > ago(7d) | summarize count() by resultSignature_d | render barchart
```



You'd run the command to see a bar chart of your search services HTTP responses. In the above, you can see there have been several 503 responses.

Check the performance of individual queries

The best way to test individual query performance is with a **client tool like Postman**.

You can use any tool that will show you the headers in the response to a query. Azure AI Search will **always return an 'elapsed-time' value** for how long it took the service to complete the query.

GET

https://.search.windows.net/indexes/test/docs?api-version=2020-06-30&search="quick brown foxes"

Send

Params

Authorization

Headers (10)

Body

Pre-request Script

Tests

Settings

Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> api-version	2020-06-30			
<input checked="" type="checkbox"/> search	"quick brown foxes"			
Key	Value	Description		

Body

Cookies

Headers (13)

Test Results

Status: 200 OK

Time: 125 ms

Size: 709 B

Save Response

KEY	VALUE
Cache-Control	no-cache
Pragma	no-cache
Content-Type	application/json; odata.metadata=minimal
Content-Encoding	gzip
Expires	-1
Vary	Accept-Encoding
request-id	4ba505e3-5d77-4702-8e4e-473a5856cb27
elapsed-time	21
OData-Version	4.0
Preference-Applied	odata.include-annotations=""
Strict-Transport-Security	max-age=15724800; includeSubDomains
Date	Thu, 04 Mar 2021 00:43:30 GMT
Content-Length	270

If you want to know how long it would take to send and then receive the response from the client, subtract the elapsed time from the total round trip. In the above, that would be 125 - 21 ms giving you 104 ms.

Optimize your index size and schema

How your search queries perform is directly connected to the size and complexity of your indexes. The smaller and more optimized your indexes, the faster Azure AI Search can respond to queries. Here are some tips that can help if you've found that you've performance issues on individual queries.

If you don't pay attention, indexes can grow over time. **You should review that all the documents in your index are still relevant and need to be searchable.**

If you can't remove any documents, can you reduce the complexity of the schema? Do you still need the same fields to be searchable? Do you still need all the skillsets you started the index with?

+ Add field	+ Add subfield	Delete					
Field name	Type	Retrievable	Filterable	Sortable	Facetable	Searchable	
HotelId	Edm.String	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
HotelName	Edm.String	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Description	Edm.String	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Description_fr	Edm.String	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Category	Edm.String	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Tags	Collection(Edm.String)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
ParkingIncluded	Edm.Boolean	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
LastRenovationDate	Edm.DateTime	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Rating	Edm.Double	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Address	Edm.ComplexType	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Location	Edm.Geography	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Rooms	Collection(Edm.String)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
id	Edm.String	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
rid	Edm.String	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

➔

+ Add field	+ Add subfield	Delete					
Field name	Type	Retrievable	Filterable	Sortable	Facetable	Searchable	
HotelId	Edm.String	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
HotelName	Edm.String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
Description	Edm.String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
Description_fr	Edm.String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
Category	Edm.String	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Tags	Collection(Edm.String)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
ParkingIncluded	Edm.Boolean	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
LastRenovationDate	Edm.DateTime	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
Rating	Edm.Double	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Address	Edm.ComplexType	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Location	Edm.Geography	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Rooms	Collection(Edm.String)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
id	Edm.String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
rid	Edm.String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Consider reviewing all the attributes you've enabled on each field. For example, adding support for filters, facets, and sorting can quadruple the storage needed to support your index.

Note: Having too many attributes on a field limits its capabilities. For example, in a field that's facetable, filterable, and searchable, you can only store 16 KB. Whereas a searchable field can hold up to 16 MB of text.

Improve the performance of your queries

If you know how the search service works, you can tune your queries to drastically improve performance. Use this checklist for writing better queries:

1. Only specify the fields you need to search using the **searchFields** parameter. As more fields require extra processing.
2. Return the **smallest number of fields** you need to render on your search results page. Returning more data takes more time.
3. Try to avoid partial search terms like **prefix search or regular expressions. These kinds of searches are more computationally expensive.**
4. Avoid using high skip values. This forces the search engine to retrieve and rank larger volumes of data.
5. Limit using facetable and filterable fields to low cardinality data.
6. Use search functions **instead of individual values in filter criteria.** For example, you can use `search.in(userid, '123,143,563,121','')` instead of `$filter=userid eq 123 or userid eq 143 or userid eq 563 or userid eq 121`.

If you've applied all of the above and still have individual queries that don't perform, you can scale out your index.

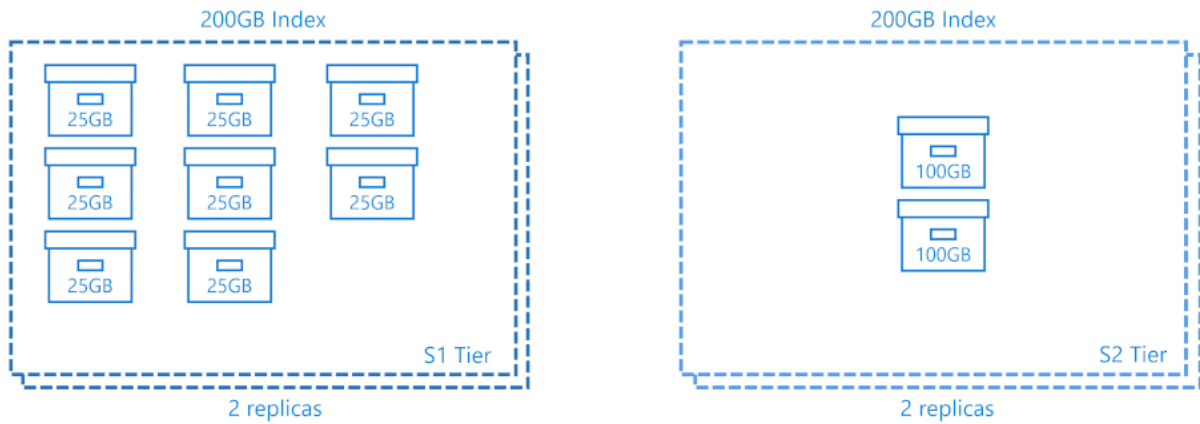
Depending on the service tier you used to create your search solution, you can add up to 12 partitions. Partitions are the physical storage where your index resides.

By default, all new search indexes are created with a single partition. If you add more partitions, the index is stored across them. For example, if your index is 200 GB and you've four partitions, each partition contains 50 GB of your index.

Adding extra partitions can help with performance as the search engine can run in parallel in each partition. The best improvements are seen for queries that return large numbers of documents and queries that use facets providing counts over large numbers of documents. This is a factor of how computationally expensive it's to score the relevancy of documents.

Use the best service tier for your search needs

You've seen that you can scale out service tiers by adding more partitions. You can scale out with replicas if you need to scale because of an increase in load. You can also *scale up* your search service by using a higher tier.



The above two search indexes are 200 GB in size. The S1 tier is using eight partitions and the S2 tier only has two. Both of them have two replicas, and both would cost approximately the same.

Choosing the best tier for your search solution requires you to know the approximate total size of storage you're going to need. The largest index supported currently is 12 partitions in the L2 tier offering a total of 24 TB.

Tier	Type	Storage	Replicas	Partitions
F	Free	50 MB	1	1
B	Basic	2 GB	3	1
S1	Standard	25 GB/Partition	12	12
S2	Standard	100 GB/Partition	12	12
S3	Standard	200 GB/Partition	12	12
S3HD	High-density	200 GB/Partition	12	3
L1	Storage Optimized	1 TB/Partition	12	12
L2	Storage Optimized	2 TB/Partition	12	12

Which of the above two tiers (S1 or S2) in the above example do you think performs the best? You've seen that scaling out gives performance benefits due to parallelism.

However, the higher tiers also come with premium storage, more powerful compute resources and extra memory. Choosing the second option gives you more powerful infrastructure and allows for future index growth.

Unfortunately which tier performs the best depends on the size and complexity of your index and the queries you write to search it. So either could be the best.

Planning for future growth in the use of your search solution means you should consider search units. A search unit (SU) is the product of replicas and partitions. That means the above S1 tier is using **16 SU** and the S2 tier is only **4 SU**. The costs are similar as higher tiers charge more per SU.

Think about needing to scale your search solution because of the increased load. Adding another replica to both tiers increase the S1 tier to **24 SU** but the S2 tier only rises to **6 SU**.

Manage costs of an Azure AI Search solution

The costs of running an Azure AI Search solution vary depending on the capacity and features you use.

Here, you'll explore the billing model, learn how to estimate baseline costs, and monitor those costs with budgets.

Estimate your search solutions baseline costs

The Azure pricing calculator is a great tool that allows you to estimate the costs of using any of the Azure services. Use it to create a baseline for your search service needs.

- 1. Browse to the [Azure AI Search pricing calculator](#).
- 2. Choose your region, currency, and hour or monthly pricing.

Region:
West US

Currency:
United States – Dollar (\$) USD

Display pricing by:
Month

	Free	Basic	Standard S1	Standard S2	Standard S3	Storage Optimised L1	Storage Optimised L2
Storage	50 MB	2 GB	25 GB (max 300 GB per service)	100 GB (max 1 TB per service)	200 GB (max 2 TB per service)	1 TB (max 12 TB per service)	2 TB (max 24 TB per service)
Max indexes per service	3	15	50	200	200 or 1000/partition in high density ¹ mode	10	10
Scale out limits	N/A	Up to 3 units per service (max 1 partition; max 3 replicas)	Up to 36 units per service (max 12 partition; max 12 replicas)	Up to 36 units per service (max 12 partition; max 12 replicas)	Up to 36 units per service (max 12 partition; max 12 replicas) up to 12 replicas in high density ¹ mode	Up to 36 units per service (max 12 partition; max 12 replicas)	Up to 36 units per service (max 12 partition; max 12 replicas) up to 12 replicas in high density ¹ mode
Price per SU (Scale Unit)	\$0/month	\$73.73/month	\$245.28/month	\$981.12/month	\$1,962.24/month	\$2,802.47/month	\$5,604.21/month

The above example shows estimates based on the number of search units. The shown monthly costs don't include everything you need for an accurate estimate. The pricing calculator also lists estimates on the additional services.

Additional Azure Cognitive Search Features (Billed Separately)

	What This Feature Does	Pricing Details
Customised Entity Lookup Skill	Looks for text from a customised, user-defined list of words and phrases and labels all documents containing matching entities. Available for all Basic, Standard, and Storage-Optimised tiers. When to use: you want to define and detect specific entities in your data.	0-1M text records \$1 per 1,000 text records 1M-3M text records \$0.75 per 1,000 text records 3M-10M text records \$0.30 per 1,000 text records 10M+ text records \$0.25 per 1,000 text records
Document Cracking: Image Extraction	Extracts content from a file within the enrichment pipeline. Text extraction is free. Image extraction is billed during the initial document cracking step and when invoking the Document Extraction skill. When to use: you have documents that contain images.	0-1M images \$1 per 1,000 transactions 1M-5M images \$0.80 per 1,000 transactions 5M+ images \$0.65 per 1,000 transactions
Semantic Search	Uses AI models to improve the relevance of the search results by finding content that is semantically similar to query terms. The service is only available for accounts on Standard tiers (S1, S2, and S3) and Storage-Optimised (L1 and L2) and has two pricing plans within those tiers. When to use: you want to improve the quality of search results and optimise the user experience.	Free Plan: Up to 1,000 semantic queries per month Standard Plan: 0-250K queries - \$500 per month; 250K + queries - \$0.002 per query

Note: The prices shown are for illustration purposes, please check the price calculator for the most up-to-date values.

Using the above information an estimate for an **S2 tier search solution, using four search units (SU)**, extracting 80,000 images, and using 200,000 semantic queries would be:

Item	Estimate
S2 tier 4SU	$\$981.12 * 4 = \mathbf{\$3,924.48}$
Cracking images	$1\$ * 80 = \mathbf{\$80}$
Semantic search	\$500
Total estimate	\$4,504.48 per month

The final costs related to running a search service are the **data ingestion and storage costs**. So the **above estimate doesn't include other infrastructure costs you can accrue**. These other costs would be things like the storage and processing of your source data.

Part of running a cost-effective Azure AI Search solution is always optimizing its capacity, from the tier you need, the data you're searching, and the features you use.

Understand the billing model

Azure AI Search is billed in the same way as other resources you use in Azure. Take the above baseline estimate as an example, after you've created the all the resources you incur costs:

- Hourly for the service tier search units you're using ($3,924.48 \div 744 = 5.27$ per hour approximately)

The other premium features are billed as you use them.

Feature	Unit
Indexer usage	Per 1000 API calls
Image extraction (AI enrichment)	Per 1000 text records

Feature	Unit
Built-in skills (AI enrichment)	Number of transactions, billed at the same rate as if you had performed the task by calling Azure AI Services directly. You can process 20 documents per indexer per day for free. Larger or more frequent workloads require a multi-resource Azure AI Services key.
Custom Entity Lookup skill (AI enrichment)	Per 1000 text records
Semantic Search	Number of queries of "queryType=semantic", billed at a progressive rate
Private Endpoints	Billed as long as the endpoint exists, and billed for bandwidth

Remember you're not charged for the number of search queries, responses, or documents ingested.

Note: There are service quotas that you should be aware of, see [Service limits in Azure AI Search](#).

Tips to reduce the cost of your search solution

These tips can help you reduce the cost of running your search solution:

1. Minimize bandwidth costs by using as few regions as possible. **Ideally, all the resources should reside in the same region.**
2. If you have predictable patterns of indexing new data, **consider scaling up inside your search tier.** Then scale back down for your regular querying.
3. To keep your search requests and responses inside the Azure datacenter boundary, use an Azure Web App front-end as your search app.
4. Enable **enrichment caching** if you're using AI enrichment on blob storage.

Manage search service costs using budgets and alerts

The most effective way to manage your costs is to monitor how much you're spending, and take action if the costs have increased over your budget.

All Azure resources can be monitored with budgets in Microsoft Cost Management. Follow the steps in [Tutorial: Create and manage Azure budgets](#) for a detailed walk-through on how to create budgets.

Home > Cost Management: Contoso (Demo)

Cost Management: Contoso (Demo) | Budgets

Billing account

Search (Ctrl+F)

Overview

Access control

Diagnose and solve problems

Cost Management

Cost analysis (preview)

Cost analysis

Cost alerts

Budgets

Advisor recommendations

Cloudyn

Billing

Usage + charges

Credits

Products + services

Azure subscriptions

Reservations

Settings

Configuration

How satisfied are you with defining and enforcing spending limits?

Scope: Contoso (Demo)

Search by name

All periods

Budget evaluations now include reserved instance and purchase charges. To learn more, visit the budgets documentation.

Name	Scope	Reset period	Creation date	Expiration date	Budget	Forecasted	Evaluated spend	Progress
EAAccount-BoTest-...	B608480 (Billing acc...	Monthly	2/1/2021	1/31/2023	\$100.00	\$0.00	\$0.00	0.00%
EAAccount-BoTest-A...	208903 (Enrollment ...	Monthly	2/1/2021	1/31/2023	\$100.00	\$0.00	\$0.00	0.00%
Pri_Forecast	B608480 (Billing acc...	Monthly	2/1/2021	2/28/2022	\$100.00	\$120.5K	\$26,046	100.00%
Pri_actualForecast	B608480 (Billing acc...	Monthly	2/1/2021	1/31/2023	\$200.00	\$120.5K	\$26,046	100.00%
EAAccount-BoTest-...	B4820 (Department)	Monthly	2/1/2021	1/31/2023	\$100.00	\$4,572	\$1,088	100.00%
Pri_EdgecaseTest	B608480 (Billing acc...	Monthly	2/1/2021	2/28/2022	\$100.00	\$515.23	\$131.43	100.00%
ACM_Department_B...	B4820 (Department)	Monthly	10/1/2019	9/30/2021	\$55,000.00	○	\$4,272	7.77%
Enrollment_budget	B608480 (Billing acc...	Monthly	4/1/2020	3/31/2022	\$45,000.00	○	\$26,046	57.88%
ACM	B608480 (Billing acc...	Monthly	8/1/2020	7/31/2022	\$30,000.00	○	\$0.00	0.00%
JoTestBudget	B608480 (Billing acc...	Monthly	11/1/2020	10/31/2022	\$50,000.00	○	\$26,046	52.09%
DemoTestBudget	B608480 (Billing acc...	Monthly	12/1/2020	11/30/2022	\$40,000.00	○	\$26,046	65.12%
ClaroTIGT	B608480 (Billing acc...	Monthly	12/1/2020	11/30/2022	\$35,000.00	○	\$26,046	74.42%

With your budget in place, you can enable alerts to notify you if your organizations search stakeholders to avoid the risks of overspending.

Improve reliability of an Azure AI Search solution

Now you've a **well-managed, secured, and cost-effective** search solution. The next step is to make sure your service is highly available and protected from disasters.

Here, you'll explore how to protect your search service reliability and make it more responsive globally.

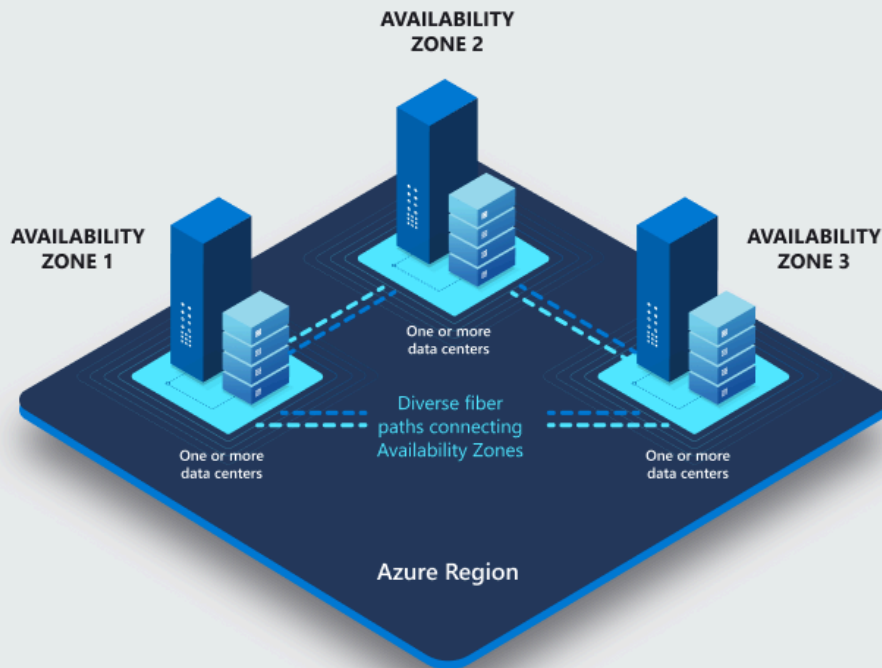
Make your search solution highly available

The first and easiest way to improve the availability of your search solution is to increase the number of replicas. The only option is to have more than one in the paid-for search service tiers.

The Azure AI Search service has availability guarantees based on the number of replicas you've:

- Two replicas guarantee **99.9% availability for your queries**
- Three or more replicas guarantee 99.9% availability for **both queries and indexing**

The second way to add redundancy to your search solution is to use the Availability Zones. This option requires that you use at least a standard tier.



When you add replicas, you can choose to host them in different Availability Zones. The benefit of distributing your replicas this way is that they're physically located in different data centers.

Distribute your search solution globally

The most cost-efficient way to architect an Azure AI Search service is in a single resource group and region (See 'Tips to reduce the cost of your search solution').

If your business priorities are availability and performance, host multiple versions of your search services in different geographical regions.

The benefits of this architecture are:

- Protection against failure in a region. Azure AI Search doesn't support instant failover, you would need to handle it manually.
- If you've globally distributed users or apps, locating a search service nearer to them will improve response times.

There's more work you'll need to do to replicate your indexes across all the regions you want to support. The options include having the same indexers based in each region ingesting the same source data. Or you can use the Push API to programmatically update all indexes in each region.

The final piece is to manage search requests through an Azure Traffic Manager to route requests to the fastest responding search index (normally this will be the **closest geographically** unless that service isn't responding).

Back up options for your search indexes

At present, Azure doesn't offer a formal backup and restore mechanism for Azure AI Search. However, you can build your own tools to back up index definitions as a series of JSON files. Then you can recreate your search indexes using these files.

Monitor an Azure AI Search solution

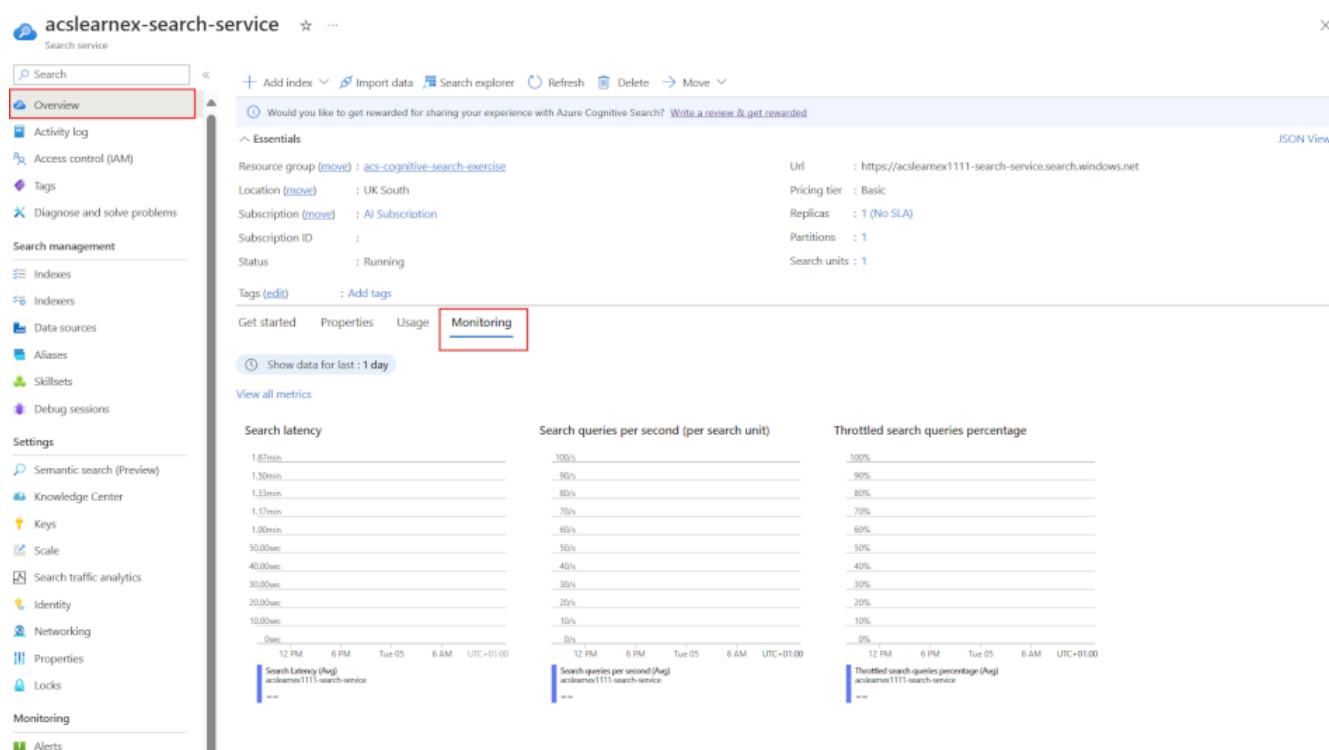
Azure Monitor can give you insights into how well your search service is being used and performing. You can also receive alerts to proactively notify you of issues.

Here, you'll explore all the monitoring options available for Azure AI Search. Then you'll learn about useful alerts you can create to manage your search solution.

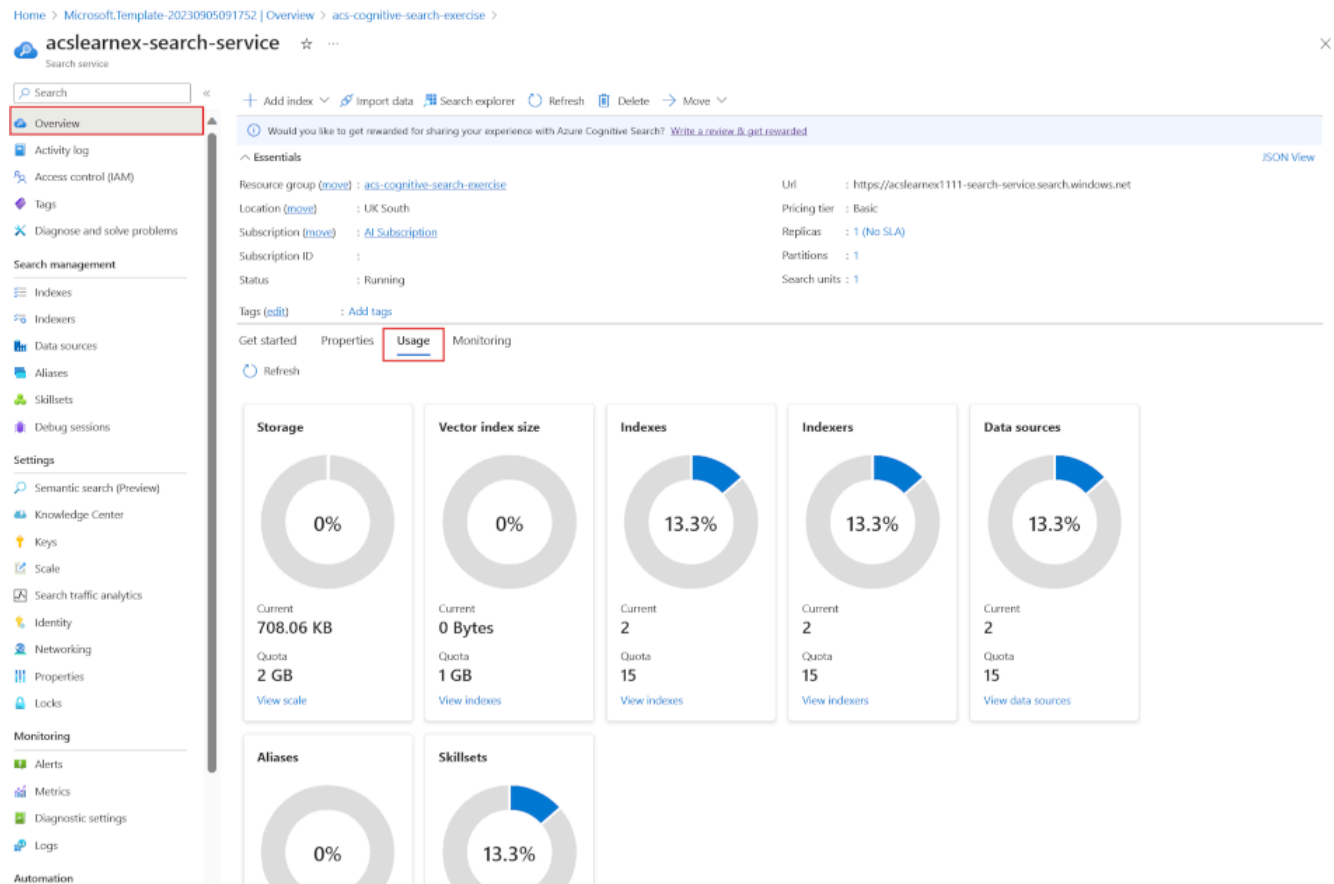
Monitor Azure AI Search in Azure Monitor

When you create your Azure AI Search service, without you doing any other setup, you can see your **current search latency, queries per second, and the percentage of throttled queries**.

This data can be viewed on the **Monitoring** tab of the **Overview** page.



You can also check what resources your search solution is using on the **Usage** tab.



This basic information is a good start to monitoring, but you can go further with some more configuration. If you're familiar with supporting other resources on the Azure platform, you'll know that Azure Monitor can be used for all your Azure resources.

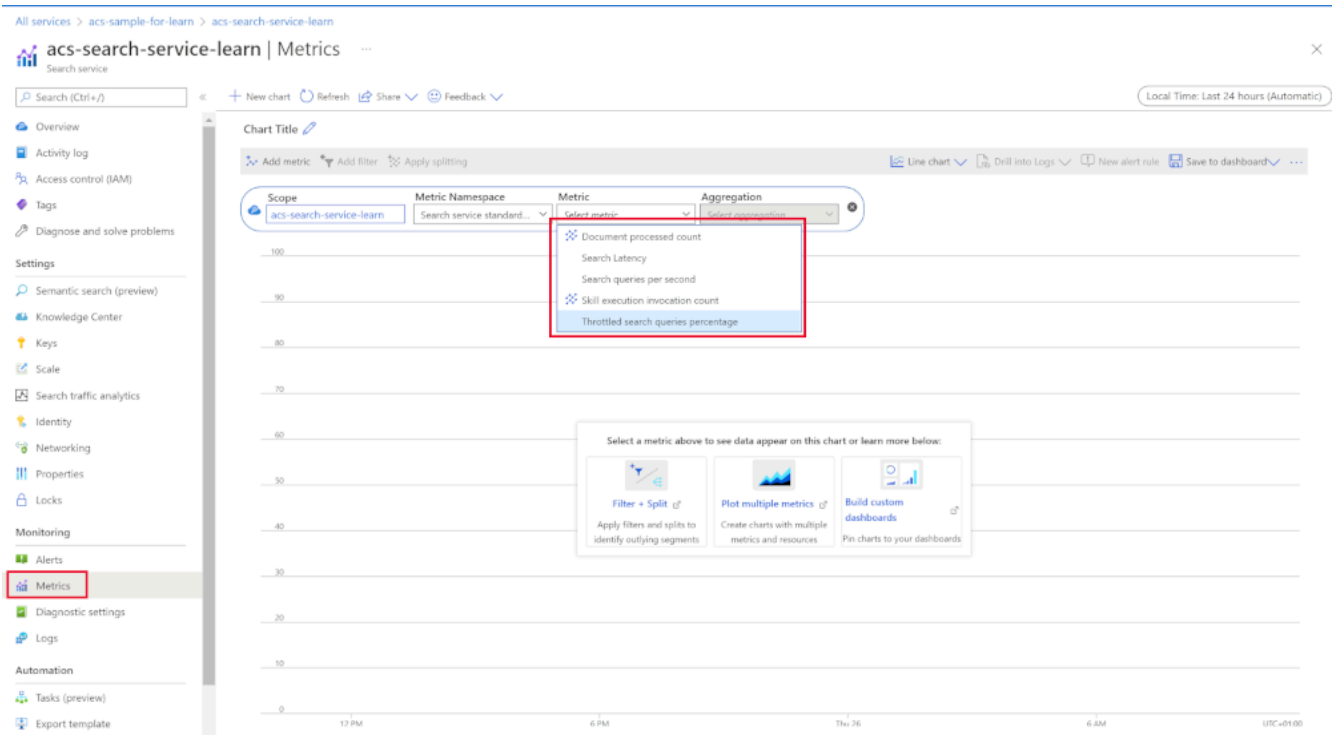
In fact, you've already seen how to enable Azure Monitor in the [optimize performance](#) unit. Follow those steps to allow Azure Monitor to use data captured in Log Analytics to see a full set of diagnostic data.

Once you have started using Log Analytics, you get access to performance and diagnostic data in these log tables:

- **AzureActivity** - Shows you tasks that have been executed like scaling the search service
- **AzureDiagnostics** - All the query and indexing operations
- **AzureMetrics** - Data used for metrics that measure the health and performance of your search service

Use metrics to see diagnostic data visually

Creating charts is a powerful way to view how your search service is performing. Under the **Monitoring** section of your search service, select **Metrics**.



Now select to add any of these captured metrics:

- DocumentsProcessedCount
- SearchLatency
- SearchQueriesPerSecond
- SkillExecutionCount
- ThrottledSearchQueriesPercentage

For example, you could plot search latency against the percentage of throttled queries to see if the responses to queries are affected by throttling.

Write Kusto queries against your search solutions logs

Log Analytics allows you to write any Kusto query against captured log data. The easiest way to run these queries is by selecting **Logs** under the Monitor section. **Logs** opens Log Analytics with the quest window automatically scoped to your Azure AI Search solution.

All services > acs-sample-for-learn > acs-search-service-learn

acs-search-service-learn

Search service

Search (Ctrl+F)

New Query 1*

acs-search-service-learn Select scope

Tables Queries Functions ...

Search

Filter Group by: Resource t...

Collapse all

Favorites

You can add favorites by clicking on the ☆ icon

Search Services

AzureDiagnostics

1 AzureDiagnostics

2 | summarize count() by OperationName

Time range: Last 24 hours

Save Share + New alert rule Export Pin to Format query

Results Chart

OperationName	count
> Indexing.Index	14
> DebugSessions.List	6
> Skillsets.List	6
> ServiceStats	11
> Indexers.Get	5
> Indexers.Status	5
>	118
> DataSources.List	11
> Query.Search	36
> Indexers.List	2
> Indexer.End	2
> Indexers.RunOnDemand	2
> Indexer.Start	2

Log

The above query lets you see a list of recent operations and how many times they happened.

```
AzureDiagnostics
| summarize count() by OperationName
```

The following are useful queries to help you monitor and diagnose issues with your search solution:

Long-running queries

```
AzureDiagnostics
| project OperationName, resultSignature_d, DurationMs, Query_s, Documents_d,
IndexName_s
| where OperationName == "Query.Search"
| sort by DurationMs
```

Indexer status

```
AzureDiagnostics
| project OperationName, Description_s, Documents_d, ResultType, resultSignature_d
| where OperationName == "Indexers.Status"
```

HTTP status codes

```
AzureDiagnostics
| where TimeGenerated > ago(7d)
| summarize count() by resultSignature_d
| render barchart
```

Query rates

```
AzureDiagnostics
| where OperationName == "Query.Search" and TimeGenerated > ago(1d)
| extend MinuteOfDay = substring(TimeGenerated, 0, 16)
| project MinuteOfDay, DurationMs, Documents_d, IndexName_s
```

```
| summarize QPM=count(), AvgDurationMs=avg(DurationMs),
AvgDocCountReturned=avg(Documents_d) by MinuteOfDay
| order by MinuteOfDay desc
| render timechart
```

Average Query Latency

```
let intervalsize = 1m;
let _startTime = datetime('2021-02-23 17:40');
let _endTime = datetime('2021-02-23 18:00');
AzureDiagnostics
| where TimeGenerated between(['_startTime']..['_endTime']) // Time range filtering
| summarize AverageQueryLatency = avgif(DurationMs, OperationName in ("Query.Search",
"Query.Suggest", "Query.Lookup", "Query.Autocomplete"))
by bin(TimeGenerated, intervalsize)
| render timechart
```

Average Queries Per Minute (QPM)

```
let intervalsize = 1m;
let _startTime = datetime('2021-02-23 17:40');
let _endTime = datetime('2021-02-23 18:00');
AzureDiagnostics
| where TimeGenerated between(['_startTime'] ..['_endTime']) // Time range filtering
| summarize QueriesPerMinute=bin(countif(OperationName in ("Query.Search",
"Query.Suggest", "Query.Lookup", "Query.Autocomplete"))/(intervalsize/1m), 0.01)
by bin(TimeGenerated, intervalsize)
| render timechart
```

Indexing Operations Per Minute (OPM)

```
let intervalsize = 1m;
let _startTime = datetime('2021-02-23 17:40');
let _endTime = datetime('2021-02-23 18:00');
AzureDiagnostics
| where TimeGenerated between(['_startTime'] ..['_endTime']) // Time range filtering
| summarize IndexingOperationsPerSecond=bin(countif(OperationName == "Indexing.Index")/
(intervalsize/1m), 0.01)
by bin(TimeGenerated, intervalsize)
| render timechart
```

Create alerts to be notified about common search solution issues

Alerts can let you proactively manage your search service. Here are some commonly used alerts you should consider creating.

- **Search Latency** using the metric signal, you can specify what latency triggers the alert in seconds
- **Throttled search percentage** using the metric signal, you can specify the percentage
- **Delete Search Service** using the activity log signal, be notified if your search service is deleted
- **Stop Search Service** using the activity log signal, be notified if your search service is stopped which happens if your search service is scaled up or down or needs to be restarted

Create an alert rule ...

Scope **Condition** Actions Details Tags Review + create

Configure when the alert rule should trigger by selecting a signal and defining its logic.

Signal name * ⓘ

Select a signal

[See all signals](#)

Select a signal



Signal type : All

Signal source : All

Signal name	Signal source
Log search	
Custom log search	Log Analytics
Resource health	
Resource health	Resource health
Metrics	
Document processed count	Platform metrics
Search Latency	Platform metrics
Search queries per second	Platform metrics
Skill execution invocation count	Platform metrics
Throttled search queries percentage	Platform metrics
Activity log	
All Administrative operations	Administrative
Approve Private Endpoint Connection (Search Services)	Administrative
Create Query Key (Search Services)	Administrative
Delete Search Service (Search Services)	Administrative
Get Admin Key (Search Services)	Administrative
Get Query Keys (Search Services)	Administrative
Regenerate Admin Key (Search Services)	Administrative
Set Search Service (Search Services)	Administrative
Start Search Service (Search Services)	Administrative
Stop Search Service (Search Services)	Administrative

Review + create

Previous

Next: Actions >

Apply

Cancel

Debug search issues using the Azure portal

When you first create your search service, you have to make some assumptions about the data you are indexing. You make choices about the index and how to ingest that data. However, until you run your created indexer you can't be certain that you made all the correct choices.

Here, you'll explore how to use the Debug Session tool inside Azure AI Search, look at debugging and then fixing a specific skill, and look at an approach to locally debugging your own custom skills.

Explore how to use the Debug Session tool in Azure AI Search

The Debug Session tool is an interactive visual editor that lets you step through the enrichment pipeline of a document as it's enriched.

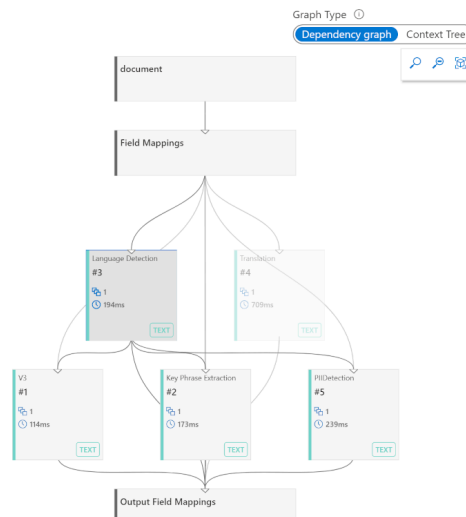
You can step into each individual skill, make changes and fixes, and then rerun the indexer in real-time. Once you've fixed any issues, you can update and republish the indexer so that it can be rerun to enrich all the documents in your index.

After you've given your debug session a name, and chosen the index you'd like to debug, the search service copies everything it needs to an Azure Storage account. The copy includes the skillset, indexer, source data, and an enriched version of the document that is in the final index.

Session status: Done
Debug session execution complete.

Definition AI Enrichments Errors/Warnings

Skill Graph Enriched Data Structure



#3

Language Detection Skill

Save Discard Delete New Skill

This skill uses a pretrained model to detect which language is used (one language ID per document). When multiple languages are used within the same text segments, the output is the LCID of the predominantly used language.

[Learn more](#)

Skill Settings Skill JSON Editor Executions Errors/Warnings

Iteration	Property	Name	Source	Path
0	INPUTS	text	Data Source	<code>*/document/Description</code>
	OUTPUTS	languageCode		<code>/document/language</code>

The session is made up of a skill graph, enriched data source, skill detail pane, execution pane, and an errors/warnings pane.

The skill detail pane allows you to expand an expression evaluator to check the value and test the inputs and outputs.

Debug a skillset with Debug Sessions

To create a Debug Session, you navigate to your search service in the Azure portal and carry out these steps:

Create a Debug Session

1. Select **Debug Sessions** under Search management in the Overview pane.
2. Select **+ Add Debug Session**.
3. In **Debug session name**, provide a name that will help you remember which skillset, indexer, and data source the debug session is about.
4. In **Storage connection string**, find a general-purpose storage account for caching the debug session.
5. In **Indexer template**, select the indexer that drives the skillset you want to debug. Copies of both the indexer and skillset are used to initialize the session.
6. In **Document to debug**, choose the first document in the index or select a specific document.

new-debug-session ...

debug session

Save Session Refresh Delete Run Cancel Commit changes...

Definition AI Enrichments Errors/Warnings

Debug session name * debug 1 ✓

Description (optional)

Storage connection string * ① DefaultEndpointsProtocol=https;AccountName=csb100320017f491c2d;AccountKey=bh2heOfw==;EndpointSuffix=core.windows.net ✓
Choose an existing connection

Managed identity authentication ① ☒ None ☐ System-assigned ☐ User-assigned

Indexer Template * ① clinical-trials-indexr
Data Source: clinical-trials-ds

Document to debug Select first document Debug specific document

7. Select **Save Session** to get started.

Explore and edit a skill

Your Debug Session lets you explore how a document is enriched as it passes through each of the AI skills. You can select a skill, review the inputs and outputs, and even see the JSON definition for the skill.

1. In the dependency graph, select a **skill**.

Dashboard > demo-search-westus2 >

cog-search-demo ...

debug session

Save Session Refresh Delete Run Commit changes...

Session status: Done
Debug session execution complete.

Definition AI Enrichments Errors/Warnings

Skill Graph Enriched Data Structure

Graph Type

Dependency graph

V3 #1

1 430ms

TEXT

Output Field Mappings

OUTPUTS

organizations /document/merged_content/organizations

locations /document/merged_content/locations

people, location, address, phone number, IP

=\$/document/merged_content/organizations

=\$/document/merged_content/locations

Expression evaluator

Context /document

Expression /document/merged_content/organizations Evaluate

Value

Wrap Columns

```
1 [
2   "Microsoft",
3   "Teams",
4   "Mesh",
5   "Accenture",
6   "Mesh for Teams",
7   "Twitter"
8 ]
```

2. In the details pane to the right, select the **Executions** tab, then in OUTPUTS, open the Expression evaluator by selecting `</>` next to **organizations**.

3. To edit the skill, select the **Skill Settings** tab.

4. Make any changes to the JSON of the skill, then select **Save**.
5. To test that the changes have fixed your issue, select **Run**.
6. If the issue is now resolved and you want to publish the changes, at the top of the pane select **Commit changes....**
7. To finish the debugging session, select **Save Session**.

Validate the field mappings

Indexers can be modified if your input data doesn't quite match the schema of your target index. Use field mappings to reshape and fix this mismatch in your data during the indexing process.

1. Select **Skill Graph**, and check that **Dependency graph** is selected.

2. Select the second step in the enrichment pipeline, **Field Mappings**.
3. Make any changes to where data should be mapped to.
4. Select **Save**.
5. Select the last step, **Output Field Mappings**.

6. Output field mappings from the skills can be fixed in the detail pane.
 7. Select **Save**.
 8. To test that the changes have fixed your issue, select **Run**.
 9. If the issue is now resolved and you want to publish the changes, at the top of the pane select **Commit changes....**
-

Exercise - Debug search issues

You've built your search solution but have noticed that there are some warnings on the indexer.

In this exercise, you'll create an Azure AI Search solution, import some sample data, and then resolve a warning on the indexer.

Note To complete this exercise, you will need a Microsoft Azure subscription. If you don't already have one, you can sign up for a free trial at <https://azure.com/free>.


Create your search solution

Before you can begin using a Debug Session, you need to create an Azure AI Search service.

1. [Deploy resources to Azure](#) - if you're in a hosted VM, copy this link and paste into the VM browser. Otherwise, select this link to deploy all the resources you need in the Azure portal.

Custom deployment ...

Deploy from a custom template

 New! Deployment Stacks let you manage the lifecycle of your deployments. Try it now →

Basics Review + create

Template



Customized template [↗](#)
11 resources


Edit template


Edit parameters


Visualize

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * ⓘ	AI Subscription ▼
Resource group * ⓘ	(New) acs-cognitive-search-exercise ▼

[Create new](#)

Instance details

Region * ⓘ	UK South ▼
Resource Prefix * ⓘ	acslearnex ✓
Storage Account Type ⓘ	Standard_LRS ▼
Hosting Plan Sku ⓘ	B1 ▼
Search Service Sku ⓘ	basic ▼
Location ⓘ	UK South ▼

Previous

Next

Review + create

- Under **Resource Group**, select your provided resource group or select **Create new** and type **debug-search-exercise**.
- Select the closest **Region** to you, or use the default.

4. For **Resource Prefix**, enter **debugsearch** and add a random combination of numbers or characters to ensure the storage name is unique.
5. For the Location, select the same region you used above.
6. At the bottom of the pane, select **Review + create**.
7. Wait until the resource is deployed, then select **Go to resource group**.

Import sample data and configure resources

With your resources created, you can now import your source data.

1. In the listed resources, navigate to the storage account. Go to **Configuration** in the left pane, set **Allow Blob anonymous access** to **Enabled** and then select **Save**.
2. Navigate back to your resource group, and select the search service.
3. On the **Overview** pane, select **Import data**.

The screenshot shows the Azure portal interface for a search service named 'acslearnex-search-service'. The left-hand navigation pane is expanded, showing various settings and management options. The 'Import data' button is highlighted with a red box in the top toolbar. The main content area displays the 'Overview' tab, which includes a list of essentials and a 'Get started' section with three cards: 'Connect your data', 'Explore your data', and 'Monitor and scale'. The 'Connect your data' card has an 'Import' button, while the other two have 'View' buttons.

Home > Microsoft.Template-20230905091752 | Overview > acs-cognitive-search-exercise >

acslearnex-search-service ☆ ...

Search service

Search

+ Add index Import data Search explorer Refresh Delete Move

Would you like to get rewarded for sharing your experience with Azure Cognitive Search? [Write a review & get rewarded](#)

JSON View

Essentials

Resource group (move) : acs-cognitive-search-exercise

Location (move) : UK South

Subscription (move) : AI Subscription

Subscription ID : 2ffdd468-1ff6-41d2-8e57-cec24b5a6553

Status : Running

Tags (edit) : Add tags

Url : https://acslearnex1111-search-service.search.windows.net

Pricing tier : Basic

Replicas : 1 (No SLA)

Partitions : 1

Search units : 1

Get started Properties Usage Monitoring

Build a full-text search experience with AI and semantic search

Get started building a full-text search experience and learn how to integrate with your custom applications and other Azure services.

Connect your data

Start here to import your data. Learn how to quickly connect to your data to build your first search index. [Learn more](#)

Import

Explore your data

Connect to apps, optimize search results. Leverage features like faceting, filtering, scoring profiles and more. [Learn more](#)

View

Monitor and scale

Tools that allow you to monitor your system and scale for optimal performance. Adjust replicas and partitions as needed. [Learn more](#)

View



4. On the import data pane, for the Data Source, select **Samples**.

Import data ...

Connect to your data Add cognitive skills (Optional) Customize target index Create an indexer

Create and load a search index using data from an external data source. Azure Cognitive Search crawls the data structure you provide, extracts searchable content, optionally enriches it with cognitive skills, and loads it into an index. [Learn more](#)


Data Source Samples

Type	Name
	realestate-us-sample
	hotels-sample

Next: Add cognitive skills (Optional)

5. In the list of samples, select **hotels-sample**.
6. Select **Next: Add cognitive skills (Optional)**.
7. Expand the **Add enrichments** section.


Import data ...

 Enrich and extract structure from your documents through cognitive skills using the same AI algorithms that power Cognitive Services. Select the document cracking options and the cognitive skills you want to apply to your documents. Optionally, save enriched documents in Azure storage for use in scenarios other than search. [Learn more](#)


▼ Attach Cognitive Services


^ Add enrichments

Run cognitive skills over a source data field to create additional searchable fields. [Learn about additional skills and extensibility here.](#)

Skillset name *  hotels-sample-skillset

Source data field * Hotelid ▼

Enrichment granularity level  Source field (default) ▼

☐ Enable incremental enrichment 

Checked items below require a field name.

<input checked="" type="checkbox"/>	Text Cognitive Skills	Parameter	Field name
<input checked="" type="checkbox"/>	Extract people names		people
<input checked="" type="checkbox"/>	Extract organization names		organizations
<input checked="" type="checkbox"/>	Extract location names		locations
<input checked="" type="checkbox"/>	Extract key phrases		keyphrases
<input checked="" type="checkbox"/>	Detect language		language ▼
<input checked="" type="checkbox"/>	Translate text	Target Language English ▼	translated_text ✓
<input checked="" type="checkbox"/>	Extract personally identifiable information		pii_entities ✓

Previous: Connect to your data

Next: Customize target index

8. Select **Text Cognitive Skills**.
9. Select **Next: Customize target index**.
10. Leave the defaults, then select **Next: Create an indexer**.
11. Select **Submit**.

Use a debug session to resolve warnings on your indexer

The indexer will now begin to ingest 50 documents. However, if you check the status of the indexer you'll find that there's warnings.

acslearnex-search-service | Indexers ☆ ...

Search service

Search

Filter by name...

+ Add indexer Refresh Delete

Status	Name	Last run	Docs succeeded	Errors/Warnings
Success	acslearnex1111-indexer	18 minutes ago	0/0	0/3
Failed	hotels-sample-indexer	1 minute ago	0/1	1/0

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Search management

Indexes

Indexers

Data sources

Aliases

Skillsets

Debug sessions

Settings

Semantic search (Preview)

Knowledge Center

Keys

Scale

1. Select **Debug sessions** in the left pane.
2. Select **+ Add Debug Session**.
3. Provide a name for the session, and select **hotel-sample-indexer** for the **Indexer Template**.
4. Select your storage account from the **Storage account** field. This will automatically create a storage container for you to hold the debug data.
5. Leave the checkbox for authenticating using a managed identity unchecked.
6. Select **Save**.
7. Once created, the debug session will automatically run on the data in your search service. It should complete with errors/warnings.

The dependency graph shows you that for each document there's an error on three skills.

Run Stop Commit changes Add new skill Undo Redo Save Refresh Settings Delete

Completed with errors/warnings. Errors: 3

React Flow

Index mapping

- Projection map
- Outputfield map
- Field mapping

Enriched data structure

```

_ts: 1725134158
Address: {}
AzureSearch_LocalDocKey: "079be536-8333-4998-75a2-20eabb48303d"
AzureSearch_PartitionKey: "\"24\""
Category: "Suite"
Description: "Chic hotel near the city. High-rise hotel in downtown, within walking distance to theaters, art gall..."
Description_fr: "Hôtel chic près de la ville. Hôtel de grande hauteur au centre-ville, à distance de marche des théât..."
HotelId: 24
HotelName: "Uptown Chic Hotel"
id: "079be536-8333-4998-75a2-20eabb48303d"
language: "(Unknown)"
LastRenovationDate: "2018-08-25T00:00:00Z"
Location: {}
ParkingIncluded: false
Rating: 3.5
rid: "z+QRAIJafXoGAAAAAAAAAA=="
Rooms: { Object[10] }
Skill_2_AdjunctOutput_languageN...: "(Unknown)"
Skill_2_AdjunctOutput_score: 0.0
  
```

Note: You may see an error about connecting to the storage account and configuring managed identities. This happens if you try to debug too quickly after enabling anonymous blob access, and running the debug session should still work. Refreshing the browser window after a few minutes should remove the warning.

8. In the dependency graph, select one of the skill nodes that have an error.
9. On the skills details pane, select **Errors/Warnings(1)**.

The details are:

Invalid language code '(Unknown)'. Supported languages:

af,am,ar,as,az,bg,bn,bs,ca,cs,cy,da,de,el,en,es,et,eu,fa,fi,fr,ga,gl,gu,he,hi,hr,hu,hy,id,it,ja,ka,kk,km,kn,ko,ku,ky,lo,lt,lv,mg,mk,ml,mn,mr,ms,my,ne,nl,no,or,pa,pl,ps,pt-BR,

PT,ro,ru,sk,sl,so,sq,sr,ss,sv,sw,ta,te,th,tr,ug,uk,ur,uz,vi,zh-Hans,zh-Hant. For additional details see <https://aka.ms/language-service/language-support>.

If you look back at the dependency graph, the Language Detection skill has outputs to the three skills with errors. If you look at the skill settings with errors, you'll see the skill input causing the error is `languageCode`.

10. In the dependency graph, select **Language detection**.

The screenshot shows the Azure Logic Apps interface. On the left, a dependency graph displays five skill nodes: #1 (V3), #2 (Key Phrase Extraction), #3 (Language Detection), #4 (Translation), and #5. Node #3 is highlighted. On the right, the 'Skill: #3' settings pane is open, showing the 'Skill Settings' tab. The JSON configuration for the skill is displayed, with the 'source' field in the 'inputs' array highlighted in red. The 'source' field is set to '/document/HotelId'.

```
1 {
2   "@odata.type": "#Microsoft.Skills.Text.LanguageDetectionSkill",
3   "name": "#3",
4   "context": "/document",
5   "inputs": [
6     {
7       "name": "text",
8       "source": "/document/HotelId",
9       "inputs": []
10    }
11  ],
12  "outputs": [
13    {
14      "name": "languageCode",
15      "targetName": "language"
16    }
17  ]
18 }
```

Looking at the skill settings JSON, note the **field being used to deduce the language is the `HotelId`**. This field will be causing the error as the skill can't work out the language based on an ID.

Resolve the warning on the indexer

1. Select **source** under inputs, and change the field to `/document/Description`.
2. Select **Save**.

3. Select **Run**. The indexer should no longer have any errors or warnings. The skillset can now be updated.

The screenshot shows the Azure AI Search Studio interface. At the top, a toolbar includes buttons for Run, Stop, Commit changes, Add new skill, Undo, Redo, Save, Refresh, Settings, and Delete. Below the toolbar, a yellow banner indicates "Completed." The main workspace displays a flow diagram with three skill nodes: #1 V3, #2 Key Phrase Extraction, and #3 Language Detection. Arrows show the flow of data between these skills. On the right, the "Enriched data structure" is shown as a JSON object with various fields including _ts, Address, AzureSearch_LocalDocKey, AzureSearch_PartitionKey, Category, Description, Description_fr, HotelId, HotelName, id, language, LastRenovationDate, Location, ParkingIncluded, Rating, rid, Rooms, Skill_2_AdjunctOutput_languageN..., Skill_2_AdjunctOutput_score, and Type.

4. Select **Commit changes** to push the changes made in this session to your indexer.
5. Select **OK**. You can now delete your session.

Now you need to make sure that your skillset is attached to an Azure AI Services resource, otherwise you'll hit the basic quota and the indexer will timeout.

1. To do this, select **Skillsets** in the left pane, then select your **hotels-sample-skillset**.

The screenshot shows the Azure AI Search Studio interface. The left pane displays the "Skillsets" section, which is highlighted with a red box. The main pane shows a table of skillsets. The table has two columns: "Name" and "Number of Skills". The "hotels-sample-skillset" is highlighted with a red box.

Name	Number of Skills
acslearnex1111-skills	7
hotels-sample-skillset	5

2. Select **Connect AI Service**, then select the AI services resource in the list.

Home > Resource groups > ResourceGroup1 > acslearnex580bc

hotels-sample-skillset

Skillset

Save Refresh Delete **Connect AI service**

Skillset Definition (JSON) Knowledge Store

```

1 {
2   "@odata.context": "https://acslearnex580bc
3   "@odata.etag": "\"0x8DC8F1E0FD04658\"",
4   "name": "hotels-sample-skillset",
5   "description": "Skillset created from the
6   "skills": [
7     {
8       "@odata.type": "#Microsoft.Skills.Text
9       "name": "#1",
10      "description": null,
11      "context": "/document/HotelId",
12      "categories": [
13        "Product",
14        "DateTime",
15        "Location",
16        "Quantity",
17        "Skill",
18        "URL",
19        "Event".

```

Connect AI service

To power your cognitive skills, select an existing AI Services resource or create a new one. The AI Services resource should be in the same region as your search service. If you decide to include cognitive skills in Azure AI Search, please note that their costs are billed separately. By choosing to add these skills, you acknowledge that you are aware of the [Pay-as-you-go pricing](#) for each skill and the [documentation that explains how each skill functions](#). [Learn more](#)

AI Services Resource Name	Region
Free (Limited enrichments)	
acslearnex580bc-cognitive-services	eastus

Refresh

[Create new AI service](#)

i REGION in the table above specifies the region only for included regional services. This does not specify a region for included non-regional services (e.g. Translator Text service). See [service status page](#) for more details.

Save

Cancel

3. Select **Save**.

4. Now run your indexer to update the documents with the fixed AI enrichments. To do this select **Indexers** in the left pane, select **hotels-sample-indexer**, then select **Run**. When it has finished running, you should see that the warnings are now zero.

acslearnex-search-service | Indexers

Search

Overview Activity log Access control (IAM) Tags Diagnose and solve problems

Search management

Indexes Indexers Data sources Aliases Skillsets Debug sessions

Filter by name...

Status	Name	Last run	Docs succeeded	Errors/Warnings
Success	acslearnex1111-indexer	1 hour ago	0/0	0/3
Success	hotels-sample-indexer	1 minute ago	33/33	0/0

Clean-up

Now you've completed the exercise, if you've finished exploring Azure AI Search services, delete the Azure resources that you created during the exercise. The easiest way to do this is delete the **debug-search-exercise** resource group.

Knowledge Check

1. An organization wants to improve the reliability of a search service. It's important that both read and write operations are 99.9% available. Which of these architectures would ensure this reliability? *

- ☐ Create an Azure AI Search service with a Storage Optimized service tier and at least two replicas.
- ☒ Create an Azure AI Search service with any Standard service tier and at least three replicas.

✓ Correct. The main factor is that the search service has three replicas.

- ☐ Create an Azure AI Search service with a High-density service tier and one replica.

2. After an Azure AI Search service has been created, which three metrics can be viewed in graphs without any other configuration? *

- ☒ Search latency, queries per second, and the percentage of throttled queries.

✓ Correct. These three metrics are graphed on the overview pane.

- ☐ Count of documents processed, count of skills executed, and the search latency.
- ☐ Number of errors per indexer, number of warnings per indexer, and the total number of documents indexed.

3. Which of the following option is the best way to manage your search service costs? *

- ☐ Enable enrichment caching if you're using AI enrichment on blob storage.
- ☐ Keep your search requests and responses inside the Azure datacenter boundary.
- ☒ Monitor and set budget alerts for all your search resources.

✓ Correct. This option is the most effective way to manage your costs.

Summary

In this module, you learned how to run and maintain a successful search solution by doing the following:

- Managing the security of your search service and source data.
- Optimizing the performance your indexes.
- Managing your costs.
- Improving reliability.
- Monitoring the performance and running queries against Log Analytics.
- Debugging indexer related errors and warnings.

For more information about Azure AI Search, take a look at the [Azure AI Search documentation](#).
