

# Azure AI 검색 설명서

기존 또는 생성 검색 시나리오에서 벡터 및 텍스트 콘텐츠에 대한 대규모 정보 검색

## Azure AI 검색 정보

### {☰} 개념

[Azure AI 검색이란?](#)

[AI 보강](#)

[의미 체계 순위 지정](#)

### 🚀 빠른 시작

[검색 서비스 만들기](#)

[검색 인덱스 만들기](#)

## RAG 및 벡터

### {☰} 개념

[Azure AI 검색의 벡터](#)

[기본 제공 벡터화](#)

[기본 제공 압축](#)

[RAG\(검색 증강 생성\)](#)

### 🚀 빠른 시작

[LLM을 사용하여 데이터 검색](#)

[벡터 인덱스 만들기](#)

[벡터 인덱스 쿼리](#)

### ⤓ SAMPLE

[벡터 샘플 ⤵](#)

## Azure AI Studio

### 방법 가이드

[AI Studio에서 벡터 인덱스 만들기](#)

[Azure OpenAI를 사용하여 데이터와 채팅](#)

[질문 및 답변 Copilot 작성](#)

## 인덱스 데이터

### 개념

[검색 인덱스란?](#)

[데이터 가져오기](#)

[인덱서 개요](#)

### 방법 가이드

[Azure Blob Storage의 인덱싱](#)

[Azure SQL Database의 인덱스](#)

[Azure Cosmos DB에서 인덱스](#)

[데이터 인덱싱](#)

## 앱 개발

### 자습서

[웹앱에 검색 추가](#)

### 방법 가이드

[.NET으로 작성된 개발](#)

[REST에서 개발](#)

### 참조

[Azure REST API 참조](#)

[Azure SDK for .NET](#)

[Python용 Azure SDK](#)

[Java용 Azure SDK](#)

[JavaScript용 Azure SDK](#)

## 쿼리 데이터

---

### 개념

[쿼리 형식 및 컴파지션](#)

[간단한 쿼리 만들기](#)

[고급 쿼리 만들기](#)

---

### 참조

[간단한 구문\(기본값\)](#)

[OData 언어 참조](#)

[문서 검색\(REST\)](#)

# Azure AI 검색이란?

아티클 • 2024. 10. 27.

Azure AI Search(이전의 "Azure Cognitive Search")는 모든 규모의 고성능 애플리케이션을 위해 빌드된 포괄적인 고급 검색 기술을 갖춘 엔터프라이즈 지원 검색 및 검색 시스템입니다.

Azure AI Search는 Azure OpenAI Service와 Azure Machine Learning 간의 네이티브 LLM 통합을 통해 Azure에서 RAG 기반 애플리케이션을 빌드할 때 권장되는 기본 검색 시스템입니다.

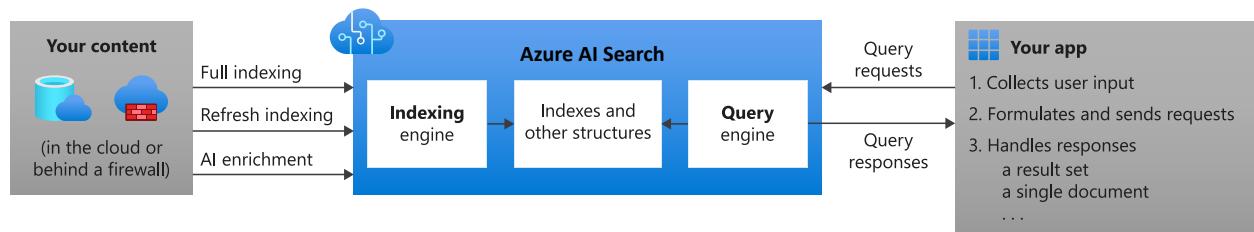
Azure AI Search는 기존 시나리오와 GenAI 시나리오 모두에서 사용할 수 있습니다. 일반적인 사용 사례로는 기술 자료 인사이트(카탈로그 또는 문서 검색), 정보 검색(데이터 탐색), RAG(검색 보강 생성) 및 자동화가 포함됩니다.

검색 서비스를 만들 때 다음 기능을 사용합니다.

- 검색 인덱스를 통한 **벡터 검색** 및 **전체 텍스트** 및 **하이브리드 검색**에 사용되는 검색 엔진
- 통합 데이터 청크 및 **벡터화**, 텍스트 어휘 분석, 콘텐츠 추출 및 변환을 위한 **선택적 응용 AI**를 통한 풍부한 인덱싱
- 벡터 쿼리**, 텍스트 검색, **하이브리드 쿼리**, 유사 항목 검색, 자동 완성, 지역 검색 등을 위한 풍부한 쿼리 구문
- 의미론적 순위 지정**, **채점 프로필**, **벡터 쿼리에 대한 양자화** 및 런타임 시 쿼리 동작을 제어하기 위한 매개 변수를 사용하여 관련성 및 쿼리 성능 조정
- Azure 규모, 보안 및 도달률
- 데이터 레이어, 기계 학습 레이어, Azure AI 서비스 및 Azure OpenAI에서 Azure 통합

## 검색 서비스 만들기

구조적으로 검색 서비스는 인덱싱되지 않은 데이터를 포함하는 외부 데이터 저장소와 검색 인덱스에 쿼리 요청을 보내고 응답을 처리하는 클라이언트 앱 사이에 배치됩니다.



클라이언트 앱에서 검색 환경은 Azure AI 검색의 API를 통해 정의되며 관련성 튜닝, 의미 체계 순위 지정, 자동 완성, 동의어 일치, 유사 일치, 패턴 일치, 필터링 및 정렬이 이 환경에 포함될 수 있습니다.

Azure 플랫폼에서 Azure AI 검색은 Azure 데이터 원본에서 데이터 수집/검색을 자동화하는 인덱서 형태, 이미지 및 자연어 처리와 같은 Azure AI 서비스의 소모성 AI 또는 Azure Machine Learning에서 만들거나 Azure Functions 내부에서 래핑하는 사용자 지정 AI를 통합하는 기술 세트 형태의 다른 Azure 서비스와 통합할 수 있습니다.

## 검색 서비스 내부

검색 서비스 자체의 두 가지 기본 워크로드는 인덱싱 및 쿼리입니다.

- **인덱싱**은 콘텐츠를 검색 서비스에 로드하여 검색 가능하게 만드는 유입 프로세스입니다. 내부적으로 인바운드 텍스트는 토큰으로 처리되고 반전된 인덱스에 저장되며 인바운드 벡터는 벡터 인덱스에 저장됩니다. Azure AI 검색에서 인덱싱할 수 있는 문서 형식은 JSON입니다. 어셈블한 JSON 문서를 업로드하거나 인덱서를 사용하여 데이터를 검색하고 JSON으로 직렬화할 수 있습니다.

**기술을 통한 적용 AI**는 이미지 및 언어 모델로 인덱싱을 확장합니다. 원본 문서에 이미지나 대규모 비정형 텍스트가 있는 경우 OCR을 수행하고, 이미지를 분석 및 설명하고, 구조를 유추하고, 텍스트를 번역하는 등의 기술을 연결할 수 있습니다. 출력은 JSON으로 직렬화하고 검색 인덱스로 수집할 수 있는 텍스트입니다.

기술 세트는 **인덱싱 중 데이터 청크 및 벡터화**도 수행할 수 있습니다. Azure AI 스튜디오의 모델 카탈로그인 Azure OpenAI에 연결하는 기술 또는 외부 청킹 및 포함 모델에 연결하는 사용자 지정 기술을 인덱싱 중에 사용하여 벡터 데이터를 만들 수 있습니다. 출력은 검색 인덱스로 수집할 수 있는 청크 분할된 벡터 콘텐츠입니다.

- 인덱스가 검색 가능한 텍스트로 채워진 경우 클라이언트 앱에서 검색 서비스에 쿼리 요청을 보내고 응답을 처리하면 **쿼리**가 발생할 수 있습니다. 모든 쿼리 실행은 사용자가 제어하는 검색 인덱스를 통해 이루어집니다.

**의미 체계 순위 지정**은 쿼리 실행 확장입니다. 결과 집합을 재평가하기 위한 언어 해석을 활용하여 보조 순위를 추가함으로써 의미론적으로 가장 관련성이 높은 결과를 맨 위로 올립니다.

**통합 벡터화**는 쿼리 실행의 확장이기도 합니다. 검색 인덱스에 벡터 필드가 있는 경우 쿼리 시 벡터화된 원시 벡터 쿼리 또는 텍스트를 제출할 수 있습니다.

## Azure AI 검색을 사용하는 이유

Azure AI 검색이 적합한 애플리케이션 시나리오는 다음과 같습니다.

- 기존 전체 텍스트 검색과 차세대 벡터 유사성 검색에 사용합니다. 키워드 및 유사성 검색의 강도를 양쪽 모두 활용하는 정보 검색을 사용하여 생성형 AI 앱을 다시 만듭니다.

니다. 두 모달리티를 모두 사용하여 가장 관련성이 큰 결과를 검색합니다.

- 이질적인 콘텐츠를 벡터와 텍스트로 구성된 사용자 정의 및 채워진 검색 인덱스에 통합합니다. 사용자는 검색 가능한 항목에 대한 소유권과 제어권을 유지합니다.
- 생성형 AI 및 RAG 앱에 대한 [데이터 청크와 벡터화](#)를 통합합니다.
- 문서 수준에서 [세분화된 액세스 제어](#)를 적용합니다.
- 인덱싱 및 쿼리 워크로드를 전용 검색 서비스로 오프로드합니다.
- 관련성 튜닝, 패싯 탐색, 필터(지리 공간 검색 포함), 동의어 매핑 및 자동 완성과 같은 관련 기능을 쉽게 구현합니다.
- Azure Blob Storage 또는 Azure Cosmos DB에 저장된 크고 구분되지 않는 텍스트, 이미지 파일 또는 애플리케이션 파일을 검색 가능한 청크로 변환합니다. 이는 Azure AI의 외부 처리를 추가하는 [AI 기술](#)을 통해 인덱싱하는 동안에 수행됩니다.
- 언어 또는 사용자 지정 텍스트 분석을 추가합니다. 비 영어 콘텐츠가 있는 경우 Azure AI 검색은 Lucene 분석기와 Microsoft 자연어 프로세서를 모두 지원합니다. 또한 분음 부호 필터링이나 문자열의 패턴 인식 및 유지와 같은 원시 콘텐츠의 특수 처리를 수행하도록 분석기를 구성할 수도 있습니다.

특정 기능에 대한 자세한 내용은 [Azure AI 검색 기능](#)을 참조하세요.

## 시작하는 방법

기능은 Azure Portal, 간단한 [REST API](#) 또는 Azure SDK(예: [.NET용 Azure SDK](#))를 통해 노출됩니다. Azure Portal은 인덱스 및 기술 집합을 프로토타이핑하고 쿼리하기 위한 도구를 사용하여 서비스 관리 및 콘텐츠 관리를 지원합니다.

## Azure Portal 사용

핵심 검색 기능에 대한 엔드투엔드 탐색은 다음 네 단계로 수행할 수 있습니다.

1. [계층](#) 및 지역을 결정합니다. 무료 Search 서비스는 구독당 하나만 허용됩니다. 무료 계층에서 모든 빠른 시작을 완료할 수 있습니다. 더 많은 용량과 기능을 사용하려면 [청구 가능한 계층](#)이 필요합니다.
2. Azure Portal에서 [검색 서비스를 만듭니다](#).
3. [데이터 가져오기 마법사로 시작합니다](#). 기본 제공 샘플 또는 지원되는 데이터 원본을 선택하여 몇 분 안에 인덱스를 만들고, 로드하고, 쿼리합니다.

4. 포털 클라이언트를 사용하여 방금 만든 검색 인덱스를 쿼리하여 **검색 탐색기로 완료**합니다.

## API 사용

또는 원자성 단계에서 검색 인덱스를 만들고, 로드하고, 쿼리할 수 있습니다.

1. 포털, [REST API](#), [.NET SDK](#) 또는 다른 SDK를 사용하여 **검색 인덱스를 만듭니다**. 인덱스 스키마는 검색 가능한 콘텐츠의 구조를 정의합니다.
2. "푸시" 모델을 사용하여 **콘텐츠를 업로드**하여 모든 소스에서 JSON 문서를 푸시하거나 소스 데이터가 **지원되는 형식**인 경우 "**풀**" 모델(인덱서)을 사용합니다.
3. 포털, [REST API](#), [.NET SDK](#) 또는 다른 SDK에서 **검색 탐색기**를 사용하여 **인덱스를 쿼리합니다**.

## 가속기 사용

또는 솔루션 가속기를 사용해 보세요.

- **데이터와의 채팅 솔루션 가속기**는 콘텐츠에 대한 사용자 지정 RAG 솔루션을 만드는 데 도움이 됩니다.
- **대화형 지식 마이닝 솔루션 가속기**는 대화형 솔루션을 만들어 연락 센터 대화록 이후 실행 가능한 인사이트를 추출하는 데 도움이 됩니다.
- **문서 지식 마이닝 가속기** 사용하면 비정형 다중 모드 문서에서 요약, 엔터티 및 메타데이터를 처리하고 추출할 수 있습니다.
- **Build your own copilot 솔루션 가속기**는 Azure OpenAI Service, Azure AI 검색 및 Microsoft Fabric을 활용하여 사용자 지정 Copilot 솔루션을 만듭니다.
  - **일반 부조종사는** 관련 문서를 식별하고, 구조화되지 않은 정보를 요약하고, 사용자 고유의 데이터를 사용하여 Word 문서 서식 파일을 생성하는 데 도움이 됩니다.
  - **Client Advisor** 올인원 사용자 지정 Copilot은 Client Advisor가 구조화된 데이터와 구조화되지 않은 데이터 모두에서 생성형 AI의 기능을 활용할 수 있도록 지원합니다. 고객이 일상적인 작업을 최적화하고 더 많은 클라이언트와 더 나은 상호 작용을 수행하도록 지원
  - **Research Assistant**는 관련 문서를 식별하고, 방대한 양의 구조화되지 않은 정보를 요약 및 분류하고, 전반적인 문서 검토 및 콘텐츠 생성을 가속화하는 자체 AI 도우미를 빌드하도록 지원합니다.

## 💡 팁

복잡한 솔루션이나 사용자 지정 솔루션에 대한 도움이 필요하면 Azure AI 검색 기술에 대해 심도 깊은 전문 지식을 갖춘 [파트너에게 문의](#)하세요.

## 검색 옵션 비교

Azure AI 검색이 다른 검색 관련 솔루션과 어떻게 비교되는지 질문하는 고객이 많습니다. 다음 표에 주요 차이점이 요약되어 있습니다.

### 테이블 확장

비교 대상	주요 차이점
Microsoft Search	<a href="#">Microsoft Search</a> 는 SharePoint의 콘텐츠를 쿼리해야 하는 Microsoft 365 인증 사용자를 위한 것입니다. Azure AI 검색은 Azure 및 모든 JSON 데이터 세트에서 콘텐츠를 끌어옵니다.
Bing	<a href="#">Bing API</a> 는 Bing.com 인덱스를 쿼리하여 용어와 일치시킵니다. Azure AI 검색은 콘텐츠로 채워진 인덱스를 검색합니다. 데이터 수집과 스키마를 제어합니다.
데이터베이스 검색	Azure SQL에는 <a href="#">전체 텍스트 검색</a> 과 <a href="#">벡터 검색</a> 이 있습니다. Azure Cosmos DB에는 <a href="#">텍스트 검색 및 벡터 검색</a> 도 있습니다. Azure AI 검색은 관련성 튜닝이나 이기종 원본의 콘텐츠와 같은 기능이 필요할 때 매력적인 대안이 됩니다. 리소스 사용률도 또 다른 변곡점입니다. 인덱싱 및 쿼리는 계산 집약적입니다. DBMS에서 검색을 오프로드하면 트랜잭션 처리에 사용되는 시스템 리소스가 보존됩니다.
전용 검색 솔루션	광범위한 기능을 제공하는 전용 검색을 결정했다고 가정하면 검색 기술 간에 최종 범주 비교가 수행됩니다. 클라우드 공급자 중에서 Azure AI 검색은 Azure의 콘텐츠를 통한 벡터, 키워드 및 하이브리드 워크로드에, 주로 정보 검색과 콘텐츠 탐색 모두를 검색하는 데 사용하는 앱에 가장 강력합니다.

주요 장점은 다음과 같습니다.

- 벡터 및 벡터가 아닌(텍스트) 인덱싱 및 쿼리를 지원합니다. 벡터 유사성 쿼리를 사용하면 쿼리가 정확히 일치하지 않더라도 쿼리와 의미 체계상 유사한 정보를 찾을 수 있습니다. 최상의 키워드 및 벡터 검색을 위해 하이브리드 검색을 사용합니다.
- 의미 체계 순위 및 점수 매기기 프로필을 통한 순위 지정 및 관련성 튜닝. 쿼리 구문은 용어 부스팅 및 필드 우선 순위 지정을 지원합니다.
- 인덱싱 계층에서의 Azure 데이터 통합(크롤러).
- 콘텐츠 텍스트와 벡터를 검색할 수 있도록 하는 변환을 위한 Azure AI 통합
- 신뢰할 수 있는 연결에 대한 Microsoft Entra 보안 및 인터넷이 없는 시나리오에서 프라이빗 연결을 위한 Azure Private Link

- **전체 검색 환경:** 56개 언어에서 언어 및 사용자 지정 텍스트 분석, 패싯, 쿼리 자동 완성 및 제안된 결과, 동의어
  - Azure 규모, 안정성 및 전체 도달률
- 

## 피드백

이 페이지가 도움이 되었나요?

Yes

No

[제품 사용자 의견 제공](#) | [Microsoft Q&A에서 도움말 보기](#)

# Azure AI Search의 새로운 기능

아티클 • 2024. 10. 16.

Azure Cognitive Search는 이제 Azure AI Search입니다. Azure AI Search 기능, 문서 및 샘플에 대한 최신 업데이트에 대해 알아봅니다.

## ① 참고

미리 보기 기능은 여기에 공지되어 있지만 한곳에서 찾을 수 있도록 [미리 보기 기능 목록](#)도 유지하고 있습니다.

## 2024년 10월

### [+] 테이블 확장

항목	Type	설명
<a href="#">Azure OpenAI에서 MRL 학습 텍스트 포함 모델에 대한 차원 요구 사항 낮추기</a>	기능	Text-embedding-3-small 및 Text-embedding-3-large는 MRL(Matryoshka Representation Learning)을 사용하여 학습됩니다. 이렇게 하면 포함 벡터를 더 적은 차원으로 잘라내고 벡터 인덱스 크기 사용량과 검색 품질 간의 균형을 조정할 수 있습니다. 2024-09-01-preview의 새로운 <code>truncationDimension</code> 기능을 사용하면 텍스트 포함 모델의 MRL 압축에 액세스할 수 있습니다. 새 벡터 필드에 대해서만 구성할 수 있습니다.
<a href="#">압축을 풀 @search.score 면 하이브리드 검색 결과에서 하위 점수 보기</a>	기능	최종 병합 및 점수 매기기 결과의 개별 쿼리 하위 점수를 확인하여 RRF(상호 순위 Fusion) 순위 결과를 조사할 수 있습니다. 새 <code>debug</code> 속성은 검색 점수의 압축을 풉니다. <code>QueryResultDocumentSubscores</code> , <code>QueryResultDocumentRerankerInput</code> 추가 <code>QueryResultDocumentSemanticField</code> 세부 정보를 제공합니다. 이러한 정의는 2024-09-01-preview에서 사용할 수 있습니다.
<a href="#">하이브리드 검색에서 벡터 쿼리에 대한 대상 필터</a>	기능	하이브리드 쿼리에 대한 필터에는 형식에 관계없이 요청에 대한 모든 하위 쿼리가 포함됩니다. 전역 필터를 재정의하여 필터 범위를 특정 하위 쿼리로 지정할 수 있습니다. 새 <code>filterOverride</code> 매개 변수는 2024-09-01-preview를 사용하여 하이브리드 쿼리에서 사용할 수 있습니다.
<a href="#">텍스트 분할 기술 (토큰 청크)</a>	응용 AI(기술)	이 기술에는 모델을 포함하기 위한 데이터 청크를 개선하는 새로운 매개 변수가 있습니다. 새 <code>unit</code> 매개 변수를 사용하면 토큰 청크를 지정할 수 있습니다. 이제 토큰 길이를 청크하여 길이를 포함 모델에 적합한 값으로 설정할 수 있습니다. 또한 데이터 청크 분할 중에 분할하지 않아야 하는 토큰 및 토큰을 지정할 수도 있습니다. 새

항목	Type	설명
		<p><a href="#">unit</a> 매개 변수 및 쿼리 하위 점수 정의는 2024-09-01-preview에서 찾을 수 있습니다.</p>
2024-09-01-preview	API	텍스트 포함-3 모델의 잘린 차원에 대한 REST API의 미리 보기 릴리스, 하이브리드 쿼리에 대한 대상 벡터 필터링, 디버깅을 위한 RRF 하위 점수 세부 정보 및 텍스트 분할 기술에 대한 토론 청크입니다.

## 2024년 8월

[+] 테이블 확장

항목	Type	설명
디버그 세션 개선 사항	기능	두 가지 중요한 개선 사항이 있습니다. 먼저 이제 통합 벡터화 및 데이터 청크 워크로드를 디버그할 수 있습니다. 둘째, 디버그 세션은 기술 및 매핑의 보다 간소화된 프레젠테이션을 위해 다시 디자인되었습니다. 흐름에서 개체를 선택하고 즉면 패널에서 세부 정보를 보거나 편집할 수 있습니다. 이전 탭 레이아웃은 페이지의 상황에 맞는 정보로 완전히 대체되었습니다.
2024-07-01	API	인덱싱 및 쿼리 중에 일반 공급되는 벡터 데이터 형식, 벡터 압축 및 통합 벡터화를 위한 REST API의 안정적인 릴리스입니다.
통합 벡터화	기능	일반 공급 발표. 인덱싱 중 기술 기반 데이터 청크 및 포함.
벡터라이저	기능	일반 공급 발표. 쿼리 실행 중 텍스트를 벡터로 변환합니다. <a href="#">Azure OpenAI 벡터화 도우미</a> 및 <a href="#">사용자 지정 웹 API 벡터화 도우미</a> 가 일반 공급됩니다.
AzureOpenAIEmbedding 기술	기능	일반 공급 발표. Azure OpenAI 포함 모델을 호출하여 인덱싱 중에 포함을 생성하는 기술 형식입니다.
지수 프로젝션	기능	일반 공급 발표. 보강 파이프라인의 콘텐츠가 여러 인덱스를 대상으로 할 수 있는 일대다 인덱스 패턴을 지원하는 보조 인덱스의 모양을 정의하는 기술 세트 정의의 구성 요소입니다.
이진 및 스칼라 양자화	기능	일반 공급 발표. 기본 제공된 양자화를 사용하여 메모리와 디스크의 벡터 인덱스 크기를 압축합니다.
좁은 데이터 형식	기능	일반 공급 발표. 들어오는 데이터가 해당 데이터 형식이라고 가정하고 벡터 필드에 더 작은 데이터 형식을 할당합니다.
데이터 가져오기 및 벡터화 마법사	Azure Portal	일반 공급 발표. 데이터 청킹 및 벡터화가 포함된 전체 인덱싱 파이프라인을 만드는 마법사입니다. 마법사는 필요한 모

항목	Type	설명
		든 개체 및 구성을 만듭니다. 이번 릴리스는 Azure Storage에 서 Azure Data Lake에 대한 마법사 지원을 추가합니다.
저장된 속성	기능	일반 공급 발표. 검색 가능한 벡터를 저장하지 않아 벡터 인덱스의 스토리지를 줄이는 부울입니다.
vectorQueries.Weight 속성	기능	일반 공급 발표. 검색 작업에서 각 벡터 쿼리의 상대적 가중치를 지정합니다.

## 2024년 7월

[+] 테이블 확장

항목	Type	설명
데이터로 채팅 ↗	액셀러레이터	Azure에서 실행되는 RAG 패턴을 위한 솔루션 가속기로, 검색을 위해 Azure AI 검색을 사용하고 대화형 검색 환경을 만들기 위해 Azure OpenAI 대규모 언어 모델을 사용합니다. 샘플 데이터가 포함된 코드는 재무 고문, 계약 검토 및 요약과 같은 사용 사례 시나리오에 사용할 수 있습니다.
대화형 지식マイ닝 ↗	액셀러레이터	Azure AI 검색, Azure Speech 및 Azure OpenAI 서비스를 기반으로 빌드된 솔루션 가속기로, 고객이 연락 센터 대화 이후 실행 가능한 인사이트를 추출할 수 있도록 지원합니다.
사용자 고유의 Copilot 빌드 ↗	액셀러레이터	Client Advisor ↗ 가 정형 및 비정형 데이터 모두에서 생성형 AI의 힘을 활용할 수 있도록 지원하는 맞춤형 Copilot 솔루션을 만드세요. 고객이 일상적인 작업을 최적화하고 더 많은 고객과 더 나은 상호 작용을 할 수 있도록 지원합니다.

## 2024년 6월

[+] 테이블 확장

항목	Type	설명
Azure Portal에서 이미지 검색	기능	이제 검색 탐색기가 이미지 검색을 지원합니다. 벡터화된 이미지 콘텐츠가 있는 벡터 인덱스에서 검색 탐색기로 이미지를 끌어와 일치 항목을 쿼리할 수 있습니다.

## 2024년 5월

[+] 테이블 확장

항목	Type	설명
모든 계층에서 더 높은 용량과 더 많은 벡터 할당량(동일한 청구 요율)	인프라	<p>대부분의 지역에서는 2024년 4월 3일 이후에 만들어진 서비스에 대해 표준 2(S2), 표준 3(S3), 표준 3 고밀도(S3 HD)의 파티션 크기가 훨씬 더 커졌습니다. 더 큰 파티션을 얻으려면 <a href="#">최신 인프라를 제공하는 지역</a>에 새 서비스를 만듭니다.</p> <p>스토리지 최적화 계층(L1 및 L2)에도 더 많은 용량이 있습니다. L1 및 L2 고객은 더 높은 용량의 이점을 활용하려면 새로운 서비스를 만들어야 합니다. 지금은 현재 위치 업그레이드가 없습니다.</p> <p>이제 다음의 <a href="#">더 많은 지역</a>에서 추가 용량 사용 가능: 독일 북부, 독일 중서부, 남아프리카 공화국 북부, 스위스 서부 및 Azure Government(텍사스, 애리조나, 버지니아).</p>
OneLake 통합( <a href="#">미리 보기</a> )	기능	<p>OneLake 파일 및 OneLake 바로 가기를 위한 새로운 인덱서. AWS(Amazon Web Services) 및 Google 데이터 원본에 대한 데이터 액세스를 위해 Microsoft Fabric 및 OneLake를 사용하는 경우 이 인덱서를 사용하여 외부 데이터를 검색 인덱스로 가져옵니다. 이 인덱서는 Azure Portal, <a href="#">2024-05-01-preview REST API</a> 및 Azure SDK 베타 패키지를 통해 사용할 수 있습니다.</p>
벡터 관련성 하이브리드 쿼리 관련성	기능	<p>네 가지 개선 사항으로 벡터 및 하이브리드 검색 관련성이 개선되었습니다.</p> <p>첫째, 이제 벡터 검색 결과에 임계값을 설정하여 점수가 낮은 결과를 제외할 수 있습니다.</p> <p>둘째, 쿼리 아키텍처의 변경 내용은 모든 쿼리 유형에 대한 쿼리 파이프라인의 끝에 채점 프로필을 적용합니다. 문서 부스팅은 일반적인 채점 프로필이며, 이제 벡터 및 하이브리드 쿼리에서 예상대로 작동합니다.</p> <p>셋째, 하이브리드 쿼리에서 <code>MaxTextRecallSize</code> 및 <code>countAndFacetMode</code>를 설정하여 하이브리드 순위 모델로 이동하는 BM25 순위 검색 결과의 수량을 제어할 수 있습니다.</p> <p>넷째, 벡터 및 하이브리드 검색의 경우 다중 쿼리 요청에서 벡터 쿼리에 가중치를 부여하여 중요도를 높이거나 낮출 수 있습니다.</p>
이진 벡터 지원	기능	<p><code>Collection(Edm.Byte)</code>는 새로 지원되는 데이터 형식입니다. 이 데이터 형식은 <a href="#">Cohere v3 이진 파일 포함 모델</a> 및 사용자 지정 이진 파일 양자화와의 통합을 제공합니다. 데이터 형식이 좁으면 대규모 벡터 데이터 세트의 비용이 낮</p>

항목	Type	설명
		아집니다. 자세한 내용은 <a href="#">벡터 검색용 이진 파일 데이터 인덱스</a> 를 참조하세요.
<a href="#">Azure AI 비전 다중 모달 포함 기술(미리 보기)</a>	기술	Azure AI 비전의 <a href="#">다중 모드 포함 API</a> 에 바인딩된 새로운 기술입니다. 인덱싱 중에 텍스트 또는 이미지에 대한 포함을 생성할 수 있습니다. 이 기술은 Azure Portal 및 <a href="#">2024-05-01-preview REST API</a> 를 통해 사용할 수 있습니다.
<a href="#">Azure AI 비전 벡터라이저(미리 보기)</a>	벡터 라이저	새로운 벡터라이저는 <a href="#">다중 모달 포함 API</a> 를 사용하여 Azure AI 비전 리소스에 연결하여 쿼리 시 포함을 생성합니다. 이 벡터라이저는 Azure Portal 및 <a href="#">2024-05-01-preview REST API</a> 를 통해 사용할 수 있습니다.
<a href="#">Azure AI 스튜디오 모델 카탈로그 벡터라이저(미리 보기)</a>	벡터 라이저	새로운 벡터라이저는 <a href="#">Azure AI 스튜디오 모델 카탈로그</a> 에서 배포된 포함 모델에 연결됩니다. 이 벡터라이저는 Azure Portal 및 <a href="#">2024-05-01-preview REST API</a> 를 통해 사용할 수 있습니다.
		<a href="#">Azure AI 스튜디오의 모델을 사용하여 통합 벡터화를 구현하는 방법</a> .
<a href="#">AzureOpenAIEmbedding 기술(미리 보기)</a> 은 Azure OpenAI에서 더 많은 모델을 지원함	기술	이제 이전 업데이트의 text-embedding-ada-002와 함께 text-embedding-3-large 및 text-embedding-3-small을 지원합니다. 새 <code>dimensions</code> 속성과 <code>modelName</code> 속성을 사용하면 Azure OpenAI에서 다양한 포함 모델을 지정할 수 있습니다. 이전에는 차원 제한이 1,536차원으로 고정되어 text-embedding-ada-002에만 적용되었습니다. 업데이트된 기술은 Azure Portal 및 <a href="#">2024-05-01-preview REST API</a> 를 통해 사용할 수 있습니다.
Azure Portal 업데이트	포털	이제 <a href="#">데이터 가져오기 및 벡터화 마법사</a> 에서 OneLake 인덱서를 데이터 원본으로 지원합니다. 포함의 경우 Azure AI Vision 멀티모달, Azure AI Studio 모델 카탈로그 및 기타 Azure OpenAI의 포함 모델에 대한 연결도 지원합니다.
		인덱스로 필드를 추가할 때 <a href="#">이진 데이터 형식</a> 을 선택할 수 있습니다.
		이제 <a href="#">검색 탐색기는</a> 기본값이 2024-05-01-preview로 설정되며 벡터 및 하이브리드 쿼리에 대한 새로운 미리 보기 기능을 지원합니다.
<a href="#">2024-05-01-preview</a>	API	Search REST API의 새로운 미리 보기 버전은 새로운 기술과 벡터라이저, 새로운 이진 데이터 형식, OneLake 파일 인덱서 및 보다 관련성이 높은 결과를 위한 새로운 쿼리 매개 변수를 제공합니다. 2023-07-01-preview에 대해 작성된 기존 코드가 있고 이 버전으로 마이그레이션해야 하는 경우 <a href="#">REST API 업그레이드</a> 를 참조하세요.

항목	Type	설명
Azure SDK 베타 패키지	API	새로운 기능 지원을 위해 다음 Azure SDK 베타 패키지의 변경 로그를 검토합니다. <a href="#">Python용 Azure SDK</a> , <a href="#">.NET용 Azure SDK</a> , <a href="#">Java용 Azure SDK</a>
Python 코드 샘플	샘플	새로운 엔드투엔드 샘플은 <a href="#">Cohere Embed v3와의 통합</a> , <a href="#">OneLake 및 Google과 AWS의 클라우드 데이터 플랫폼과의 통합</a> , <a href="#">Azure AI 비전 멀티모달 API와의 통합</a> 을 보여 줍니다.

## 2024년 4월

[+] 테이블 확장

항목	Type	설명
<a href="#">정보 공개를 해결하는 보안 업데이트</a>	API	GET 응답은 더 이상 연결 문자열이나 키를 반환하지 않습니다. GET Skillset, GET Index 및 GET Indexer에 적용됩니다. 이 변경은 AI Search와 통합된 Azure 자산을 무단 액세스로부터 보호하는 데 도움이 됩니다.
<a href="#">기본 및 표준 계층의 추가 스토리지</a>	인프라	이제 기본은 최대 3개의 파티션과 3개의 복제본을 지원합니다. 기본 및 표준(S1, S2, S3) 계층에는 파티션당 동일한 파티션당 청구 속도로 훨씬 더 많은 스토리지가 있습니다. 추가 용량은 <a href="#">지역 가용성</a> 이 적용되며 2024년 4월 3일 이후에 생성된 새 검색 서비스에 적용됩니다. 현재는 업그레이드가 없으므로 추가 스토리지를 확보하려면 새 검색 서비스를 만들어야 합니다.
<a href="#">벡터에 대한 추가 할당량</a>	인프라	선택한 지역에서 2024년 4월 3일 이후에 생성된 새 서비스에서도 벡터 할당량이 더 높습니다.
<a href="#">벡터 양자화, 좁은 벡터 데이터 형식 및 새로운 stored 속성(미리 보기)</a>	기능	종합적으로 이 세 가지 기능은 벡터 압축과 더욱 스마트한 저장 옵션을 추가합니다. 먼저 스칼라 양자화 메모리 및 디스크의 벡터 인덱스 크기를 줄입니다. 둘째, <a href="#">좁은 데이터 형식</a> 은 더 작은 값을 저장하여 필드당 스토리지를 줄입니다. 셋째, <a href="#">stored</a> 를 사용하여 검색 결과에만 사용되는 벡터의 추가 복사본 저장을 선택 해제할 수 있습니다. 쿼리 응답에 벡터가 필요하지 않은 경우 <a href="#">stored</a> 를 false로 설정하여 공간을 절약할 수 있습니다.
<a href="#">2024-03-01-preview 검색 REST API</a>	API	새 데이터 형식, 벡터 압축 속성, 벡터 스토리지 옵션에 대한 검색 REST API의 새 미리 보기 버전입니다.
<a href="#">2024-03-01-preview 관리 REST API</a>	API	컨트롤 플레인 작업에 대한 관리 REST API의 새 미리 보기 버전입니다.
<a href="#">2023-07-01-preview 사용 중</a>	API	2024년 4월 8일에 사용 중단이 공지되었습니다. 2024년 7월 8일에 지원되지 않습니다. 이는 벡터 검색 지원을 제공하는 최초의 REST API였습

항목	Type	설명
단 공지		니다. 최신 API 버전에는 벡터 구성이 다릅니다. 가능한 한 빨리 <a href="#">최신 버전으로 마이그레이션</a> 해야 합니다.

## 2024년 2월

[+] 테이블 확장

항목	Type	설명
새 차원 제한	기능	벡터 필드의 경우 이제 최대 차원 제한이 2048에서 3072로 증가했습니다.

## 2023 알림

[+] 테이블 확장

Month	Type	알림
11월	기능	<a href="#">벡터 검색(일반 공급)</a> . 이제 CMK(고객 관리형 키)에 대한 이전 제한이 해제되었습니다. 이제 <a href="#">사전 필터링 및 전체 K-가장 인접한 항목 알고리즘</a> 도 일반 공급됩니다.
11월	기능	<a href="#">의미 순위매기기, 일반 공급</a>
11월	기능	<a href="#">통합 벡터화(미리 보기)</a> 는 인덱싱 중에 데이터 청크 및 텍스트-벡터 변환을 추가하고 쿼리 시 텍스트-벡터 변환도 추가합니다.
11월	기능	<a href="#">데이터 가져오기 및 벡터화 마법사(미리 보기)</a> 는 자동으로 데이터 청크 및 벡터화합니다. 이는 2023-10-01-Preview REST API를 대상으로 합니다.
11월	기능	<a href="#">인덱스 프로젝션(미리 보기)</a> 은 보강 파이프라인의 콘텐츠가 여러 인덱스를 대상으로 할 수 있는 일대다 인덱스 패턴에 사용되는 보조 인덱스의 형태를 정의합니다.
11월	API	<a href="#">2023-11-01 검색 REST API</a> 는 벡터 검색 및 의미 순위 지정을 위한 안정적인 검색 REST API 버전입니다. 일반 공급 기능으로의 마이그레이션 단계는 <a href="#">REST API 업그레이드</a> 를 참조하세요.
11월	API	<a href="#">2023-11-01 Management REST API</a> 에는 의미 순위매기기를 사용하거나 사용하지 않도록 설정하는 API가 추가되었습니다.
11월	기술	<a href="#">Azure OpenAI Embedding 기술(미리 보기)</a> 은 Azure OpenAI 리소스에 배포된 포함 모델에 연결하여 기술 세트 실행 중에 포함을 생성합니다.
11월	기술	<a href="#">텍스트 분할 기술(미리 보기)</a> 은 네이티브 데이터 청크를 지원하도록 2023-10-01-Preview에서 업데이트되었습니다.

Month	Type	알림
11월	동영상	<b>벡터 검색 및 의미 순위 지정이 GPT 프롬프트를 개선하는 방법</b> <a href="#">☞</a> 에서는 하이브리드 검색이 유용한 AI 응답을 생성하기 위한 최적의 기반 데이터를 제공하고 개념과 키워드 모두에 대한 검색을 지원하는 방법을 설명합니다.
11월	예제	<b>생성형 AI 애플리케이션의 역할 기반 액세스 제어</b> <a href="#">☞</a> 에서는 Microsoft Entra ID 및 Microsoft Graph API를 사용하여 인덱스에 있는 청크 분할 콘텐츠에 대한 세분화된 사용자 권한을 적용하는 방법을 설명합니다.
10월	예제	"데이터와 채팅" 솔루션 가속기 <a href="#">☞</a> . Azure AI Search를 검색기로 사용하는 엔드투엔드 RAG 패턴입니다. 인덱싱, 데이터 청크 및 오케스트레이션을 제공합니다.
10월	기능	벡터 공간에서 유사성 검색을 위한 <b>Exhaust KNN(K-가장 인접한 항목)</b> 채점 알고리즘입니다. 2023-10-01-Preview REST API에서만 사용할 수 있습니다.
10월	기능	<b>벡터 쿼리의 사전 필터</b> 는 쿼리 실행 전에 필터 조건을 평가하여 쿼리해야 하는 콘텐츠의 양을 줄입니다. 요구 사항에 따라 <code>prefilter</code> (기본값) 또는 <code>postFilter</code> 으로 설정될 수 있는 새 <code>vectorFilterMode</code> 속성을 통해 2023-10-01-Preview REST API에서만 사용할 수 있습니다.
10월	API	<b>2023-10-01-Preview Search REST API</b> , 벡터 필드 및 <b>벡터 쿼리</b> 에 대한 정의가 변경되었습니다.
8월	기능	<b>향상된 의미 순위 지정</b> . 업그레이드된 모델은 의미 체계 순서 재지정을 위해 룰아웃되고 있으며 가용성이 더 많은 지역으로 확장되고 있습니다. 최대 고유 토큰 수는 128개에서 256개로 두 배 증가했습니다.
7월	예제	<b>벡터 데모(JavaScript용 Azure SDK)</b> <a href="#">☞</a> . Node.js 및 <code>@azure/search-documents 12.0.0-beta.2</code> 라이브러리를 사용하여 포함을 생성하고 인덱스를 만들고 로드하며 여러 벡터 쿼리를 실행합니다.
7월	예제	<b>벡터 데모(.NET용 Azure SDK)</b> <a href="#">☞</a> . <code>Azure.Search.Documents 11.5.0-beta.3</code> 라이브러리를 사용하여 포함을 생성하고 인덱스를 만들고 로드하며 여러 벡터 쿼리를 실행합니다. Azure SDK 팀의 <a href="#">이 샘플</a> <a href="#">☞</a> 을 사용해 볼 수도 있습니다.
7월	예제	<b>벡터 데모(Python용 Azure SDK)</b> <a href="#">☞</a> <code>azure.search.documents</code> 의 최신 베타 릴리스를 사용하여 포함을 생성하고 인덱스를 만들고 로드하며 여러 벡터 쿼리를 실행합니다. 추가 벡터 검색 데모에 대해서는 <a href="#">azure-search-vector-samples/demo-python</a> <a href="#">☞</a> 리포지토리를 방문하세요.
6월	기능	<b>벡터 검색 공개 미리 보기</b> .
6월	기능	<b>의미 체계 검색 가용성</b> (기본 계층에서 사용 가능)
6월	API	<b>2023-07-01-Preview 검색 REST API</b> . 벡터 검색을 지원합니다.
5월	기능	<b>Azure RBAC(역할 기반 액세스 제어), 일반 공급</b> .
5월	API	<b>2022-09-01 관리 REST API</b> - Azure 역할을 사용하도록 검색 구성을 지원합니다. Azure PowerShell의 <code>Az.Search</code> 모듈 및 Azure CLI의 <code>Az search</code> 모듈은 검색 서비스 인증 옵션을 지원하도록 업데이트되었습니다. <a href="#">Terraform 공급자</a> <a href="#">☞</a> 를 사용하

Month	Type	알림
		여 인증 옵션을 구성할 수도 있습니다(자세한 내용은 이 <a href="#">Terraform 빠른 시작 참조</a> ).
April	예제	<b>비즈니스 연속성 및 재해 복구를 위한 Azure AI 검색의 다중 지역 배포</b> . 엔드포인트가 실패할 경우 콘텐츠 동기화 및 요청 리디렉션 옵션을 사용하여 Azure AI Search에 대한 다중 지역 솔루션을 완전히 구성하는 배포 스크립트입니다.
March	예제	<b>Azure OpenAI 및 Azure AI 검색(GitHub)를 사용한 ChatGPT + Enterprise 데이터</b> . Azure AI Search를 OpenAI의 대규모 언어 모델과 결합하기 위한 Python 코드 및 템플릿입니다. 백그라운드 정보는 기술 커뮤니티 블로그 게시물 <a href="#">ChatGPT로 기업 데이터 혁신</a> 을 참조하세요.

핵심 내용:

Azure AI Search를 사용하여 검색 가능한 콘텐츠를 통합하고 인덱스를 생성합니다.

초기 검색 결과에 대한 인덱스를 쿼리합니다.

해당 결과에서 프롬프트를 조합하고 Azure OpenAI의 gpt-35-turbo(미리 보기) 모델로 보냅니다.

사용자가 답변을 평가할 수 있도록 문서 간 답변을 반환하고 고객 대면 앱에서 인용 및 투명성을 제공합니다.

## 작년 알림

- [2022 알림](#)
- [2021 알림](#)
- [2020 알림](#)
- [2019 알림](#)

## 서비스 리브랜딩

이 서비스는 수년에 걸쳐 여러 이름으로 지칭되었습니다. 여기서는 역순으로 표시됩니다.

- **Azure AI Search**(2023년 11월) Azure AI 서비스 및 고객의 기대에 맞게 이름이 변경되었습니다.
- **Azure Cognitive Search**(2019년 10월) 서비스 작업에서 인식 기술 및 AI 처리의 확장된 사용(아직 선택 사항임)을 반영하기 위해 이름이 변경되었습니다.
- **Azure Search**(2015년 3월) 원래 이름입니다.

# 서비스 업데이트

Azure AI Search에 대한 [서비스 업데이트 공지](#)는 Azure 웹 사이트에서 확인할 수 있습니다.

## 기능 이름 바꾸기

의미 체계 검색은 기존 결과 집합의 L2 순위를 제공하는 기능을 더 잘 설명하기 위해 2023년 11월에 [의미 순위매기기](#)로 이름이 바뀌었습니다.

## 피드백

이 페이지가 도움이 되었나요?

 Yes

 No

[제품 사용자 의견 제공](#) | [Microsoft Q&A에서 도움말 보기](#)

# Azure AI 검색의 기능

아티클 • 2024. 09. 03.

Azure AI 검색은 정보 검색을 제공하고 선택적 AI 통합을 사용하여 더 많은 텍스트 및 구조 콘텐츠를 추출합니다.

다음 표에는 범주별로 해당 기능이 요약되어 있습니다. Azure AI 검색을 다른 검색 기술과 비교하는 방법에 대한 자세한 내용은 [검색 옵션 비교](#)를 참조하세요.

모든 Azure 퍼블릭, 프라이빗 및 소버린 클라우드에는 기능 패리티가 있지만 일부 기능은 특정 Azure 지역에서 지원되지 않습니다. 자세한 내용은 [지역 선택](#)을 참조하세요.

## ① 참고

미리 보기 기능을 찾고 계신가요? [미리 보기 기능 목록](#)을 참조하세요.

## 인덱싱 기능

[\[+\] 테이블 확장](#)

범주	기능
데이터 원본	JSON 문서로 제출된 경우 검색 인덱스에서 모든 원본의 텍스트를 허용할 수 있습니다. <b>인덱서</b> 는 지원되는 데이터 원본에서 데이터 가져오기를 자동화하여 기본 데이터 저장소에서 검색 가능한 콘텐츠를 추출하는 기능입니다. 인덱서는 JSON serialization을 처리하며 대부분은 일부 형태의 변경 및 삭제 검색을 지원합니다. <a href="#">OneLake</a> , <a href="#">Azure SQL Database</a> , <a href="#">Azure Cosmos DB</a> 또는 <a href="#">Azure Blob storage</a> 를 비롯한 다양한 데이터 원본에 연결할 수 있습니다.
계층형 및 중첩 데이터 구조	복합 형식 및 컬렉션을 사용하면 거의 모든 유형의 JSON 구조를 검색 인덱스로 모델링할 수 있습니다. 일대다 및 다대다 카디널리티는 컬렉션, 복합 형식 및 복합 형식 컬렉션을 통해 고유하게 표현할 수 있습니다.
언어 분석	분석기는 인덱싱 및 검색 작업 중 텍스트 처리에 사용되는 구성 요소입니다. 기본적으로, 범용 표준 Lucene 분석기를 사용하거나, 언어 분석기, 구성한 사용자 지정 분석기 또는 필요한 형식으로 토큰을 생성하는 미리 정의된 다른 분석기를 사용하여 기본값을 재정의할 수 있습니다. Lucene 또는 Microsoft의 언어 분석기는 동사 시제, 성, 불규칙 복수 명사(예: 'mouse'와 'mice'), 단어 분해, 단어 분절(띄어쓰기가 없는 언어의 경우) 등을 비롯한 언어별 언어 체계를 지능적으로 처리할 수 있습니다.

범주	기능
	<b>사용자 지정 어휘 분석기</b> 는 표기 일치 및 정규식을 사용하는 복잡한 검색 쿼리에 사용됩니다.

## 벡터 및 하이브리드 검색

[+] 테이블 확장

범주	기능
벡터 인덱싱	검색 인덱스 내에서 <b>벡터 필드</b> 를 추가하여 <b>벡터 검색</b> 시나리오를 지원합니다. 벡터 필드는 동일한 검색 문서의 비벡터 필드와 공존할 수 있습니다.
벡터 쿼리	<b>단일 및 복수 벡터 쿼리를 작성합니다.</b>
벡터 검색 알고리즘	검색 인덱스에서 유사한 벡터를 찾으려면 <a href="#">HNSW(Hierarchical Navigable Small World)</a> 를 사용하거나 <a href="#">완전 KNN(K-Nearest Neighbors)</a> 을 사용합니다.
벡터 필터	정보 검색 시 정밀도를 높이기 위해 <b>쿼리 실행 전후에 필터를 적용합니다.</b>
하이브리드 정보 검색	단일 <a href="#">하이브리드 쿼리 요청</a> 개념 및 키워드를 검색합니다. <b>하이브리드 검색</b> 은 최상의 결과를 위해 선택적 의미 체계 순위 지정 및 관련성 튜닝과 함께 벡터 및 텍스트 검색을 통합합니다.
통합 데이터 청크 및 벡터화(미리 보기)	<a href="#">텍스트 분할 기술</a> 을 통한 원시 데이터 청크. <b>벡터라이저</b> 를 통한 원시 벡터화 및 <a href="#">AzureOpenAIEmbeddingModel</a> , <a href="#">Azure AI Vision</a> 다중 모달, <a href="#">AML</a> 기술 등 <a href="#">Azure AI Studio</a> 모델 카탈로그의 엔드포인트에 연결할 때 사용할 수 있는 포함된 기술. <b>통합 벡터화(미리 보기)</b> 는 원본 파일에서 쿼리로의 엔드투엔드 인덱싱 파이프라인을 제공합니다.
통합 벡터 압축 및 양자화	<a href="#">기본 제공 스칼라 양자화</a> 를 사용하여 메모리 및 디스크의 벡터 인덱스 크기를 줄입니다. 필요하지 않은 벡터의 스토리지를 포기하거나 스토리지 요구 사항을 줄이기 위해 벡터 필드에 좁은 데이터 형식을 할당할 수도 있습니다.

## 응용 AI 및 정보 마이닝

[+] 테이블 확장

범주	기능
인덱싱 중 AI 처리	<b>AI 보강</b> 은 전체 텍스트 검색을 위해 인덱싱할 수 없는 콘텐츠에서 텍스트 및 정보를 추출하는 인덱서 파이프라인에 포함된 이미지 및 자연어 처리를 나타냅니다. AI 처리는 기술 세트에서 기술을 추가하고 결합한 다음 인덱서에 연결하여 수행

범주	기능
	됩니다. AI는 텍스트 번역이나 OCR(광학 문자 인식)과 같은 <a href="#">기본 제공 기술</a> 이나 사용자가 제공하는 <a href="#">사용자 지정 기술</a> 일 수 있습니다.
비검색 시나리오에서 분석하고 사용할 수 있도록 보강된 콘텐츠 저장	<b>지식 저장소</b> 는 지식 마이닝 및 데이터 과학 처리와 같은 비검색 시나리오를 위한 보강된 콘텐츠의 영구적 스토리지입니다. 지식 저장소는 기술 세트에 정의되어 있지만 Azure Storage에서 개체 또는 테이블 형식 행 집합으로 만들어집니다.
캐시된 보강	<b>증분 보강(미리 보기)</b> 은 기술 세트 실행 중에 다시 사용할 수 있는 캐시된 보강을 나타냅니다. 캐싱은 처리 비용이 많이 드는 OCR 및 이미지 분석을 포함하는 기술 세트에서 특히 유용합니다.

## 전체 텍스트 및 기타 쿼리 양식

[+] 테이블 확장

범주	기능
자유 형식 텍스트 검색	<b>전체 텍스트 검색</b> 은 대부분의 검색 기반 앱에서 기본적으로 사용되는 검색 방식입니다. 쿼리는 지원되는 구문을 사용하여 작성할 수 있습니다.
간단한 쿼리 구문	<b>간단한 쿼리 구문</b> 은 논리 연산자, 구 검색 연산자, 후위 연산자, 선행 연산자를 제공합니다.
전체 Lucene 쿼리 구문	<b>전체 Lucene 쿼리 구문</b> 에는 간단한 구문의 모든 연산이 포함되며 유사 항목 검색, 근접 검색, 용어 상승 및 정규식에 대한 확장이 포함됩니다.
간단한 점수 매기기	<b>간단한 점수 매기기</b> 는 Azure AI Search의 핵심적인 장점입니다. 점수 매기기 프로필은 문서 자체의 값에 대한 함수로서 관련성을 모델링하는 데 사용됩니다. 예를 들어 최신 제품 또는 할인 제품을 검색 결과의 더 높은 부분에 나타내려고 할 수 있습니다. 또한 따로 추적하고 저장한 고객 검색 기본 설정에 따라 개인화된 점수에 태그를 사용하여 점수 매기기 프로필을 만들 수도 있습니다.
의미 순위매기기(미리 보기)	<b>의미 순위매기기(미리 보기)</b> 는 쿼리에 대한 의미 체계 관련성에 따라 결과 순위를 재조정하는 프리미엄 기능입니다. 콘텐츠 및 시나리오에 따라 거의 최소한의 구성이나 노력으로 검색 관련성을 크게 향상시킬 수 있습니다.
지리 공간적 검색	<b>지리 공간적 함수</b> 는 지리적 좌표를 필터링하고 일치시킵니다. <a href="#">거리를 일치시키거나 다각형 도형에 포함하여 일치시킬 수 있습니다.</a>
패싯 탐색	<b>패싯 탐색</b> 은 단일 쿼리 매개 변수를 통해 사용하도록 설정됩니다. Azure AI 검색은 자기 주도형 필터링(예: 가격 범위 또는 브랜드별로 카탈로그 항목 필터링)을 위해 코드 숨김 카탈로그 목록으로 사용할 수 있는 패싯 탐색 구조를 반환합니다.
필터	<b>필터</b> 를 사용하여 패싯 탐색을 애플리케이션 UI에 통합하고, 쿼리 작성률을 향상하고, 사용자

범주	기능
	또는 개발자가 지정한 기준에 따라 필터링할 수 있습니다. OData 구문을 사용하여 필터를 만듭니다.
사용자 환경	<p><b>자동 완성</b>은 검색 창의 자동 완성 쿼리에 사용할 수 있습니다.</p> <p><b>검색 제안</b>은 검색 표시줄 결과에서 일부 텍스트 입력에서 작동에 실제 문서 쿼리를 사용하지 않고 인덱스 사용 약관이 있습니다.</p> <p><b>동의어</b>는 사용자가 대체 용어를 제공할 필요 없이 쿼리의 범위를 암시적으로 확장하는 동등한 용어를 연결합니다.</p> <p><b>적중 항목 강조 표시</b>는 검색 결과의 일치하는 키워드에 텍스트 서식을 적용합니다. 강조 표시된 조각을 반환할 필드를 선택할 수 있습니다.</p> <p><b>정렬</b>은 인덱스 스키마를 통해 여러 필드에 제공된 후 쿼리 시 단일 검색 매개 변수를 통해 전환됩니다.</p> <p>검색 결과 <b>페이지</b> 및 제한은 Azure AI 검색에서 검색 결과에 대해 제공하는 세밀하게 조정된 컨트롤을 사용하여 손쉽게 적용할 수 있습니다.</p>
	<span>[+] 테이블 확장</span>

## 보안 기능

범주	기능
데이터 암호화	<p><b>Microsoft 관리형 저장 데이터 암호화</b>는 내부 스토리지 레이어에 내장되어 있으며 취소할 수 없습니다.</p> <p>Azure Key Vault에서 만들고 관리하는 <b>고객 관리형 암호화 키</b>는 인덱스 및 동의어 맵의 보충 암호화에 사용할 수 있습니다. 2020년 8월 1일 이후에 만들어진 서비스의 경우 CMK 암호화는 인덱싱된 콘텐츠의 전체 이중 암호화를 위해 임시 디스크의 데이터까지 확장됩니다.</p>
엔드포인트 보호	<p><b>인바운드 방화벽 지원을 위한 IP 규칙</b>을 사용하면 검색 서비스에서 요청을 수락하는 IP 범위를 설정할 수 있습니다.</p> <p>Azure 프라이빗 링크를 사용하여 <b>프라이빗 엔드포인트를 만들어</b> 가상 네트워크를 통해 모든 요청을 강제로 적용합니다.</p>
인바운드 액세스	<p><b>역할 기반 액세스 제어</b>는 검색 콘텐츠 및 작업에 대한 제어된 액세스를 위해 Microsoft Entra ID의 사용자 및 그룹에 역할을 할당합니다. 역할 할당이 없는 경우 <b>키 기반 인증</b>을 사용할 수도 있습니다.</p>

범주	기능
아웃바운드 보안 (인덱서)	<p><a href="#">프라이빗 엔드포인트를 통한 데이터 액세스</a>를 사용하면 인덱서에서 Azure Private Link를 통해 보호되는 Azure 리소스에 연결할 수 있습니다.</p> <p><a href="#">신뢰할 수 있는 ID를 사용하는 데이터 액세스</a>는 외부 데이터 원본에 대한 연결 문자열에서 사용자 이름과 암호를 생략할 수 있음을 의미합니다. 인덱서가 데이터 원본에 연결되면 검색 서비스가 이전에 신뢰할 수 있는 서비스로 등록된 경우 리소스에서 연결을 허용합니다.</p>

## 포털 기능

[+] [테이블 확장](#)

범주	기능
프로토타입 및 검사 용 도구	<p><a href="#">인덱스 추가</a>는 특성 사용 필드와 몇 가지 다른 설정으로 구성된 기본 스키마를 만드는데 사용할 수 있는 포털의 인덱스 디자이너입니다. 인덱스를 저장한 후에는 SDK 또는 REST API를 사용하여 데이터를 제공하는 방법으로 작성할 수 있습니다.</p> <p><a href="#">데이터 가져오기 마법사</a>는 인덱스, 인덱서, 기술 세트와 데이터 원본 정의를 만듭니다. 데이터가 Azure에 있는 경우, 이 마법사는 특히 개념 증명 조사 및 탐색에 대해 상당한 시간과 노력을 절감할 수 있습니다.</p> <p><a href="#">데이터 가져오기 및 벡터화</a>는 데이터 청킹 및 벡터화가 포함된 전체 인덱싱 파이프라인을 만듭니다. 마법사는 모든 개체 및 구성 설정을 만듭니다.</p> <p><a href="#">검색 탐색기</a>는 쿼리를 테스트하고 점수 매기기 프로필을 구체화하는 데 사용됩니다.</p> <p><a href="#">데모 앱 만들기</a>는 검색 환경을 테스트하는 데 사용할 수 있는 HTML 페이지를 생성하는 데 사용됩니다.</p> <p><a href="#">디버그 세션</a>은 대화형으로 기술 세트를 디버그할 수 있는 시각적 편집기입니다. 이것은 종속성, 출력 및 변환을 보여 줍니다.</p>
모니터링 및 진단	포털에 항상 표시되는 단순한 메트릭 이상을 원하신다면 <a href="#">모니터링 기능을 사용하세요</a> . 필요한 추가 구성 없이 포털 페이지에서 초당 쿼리 수, 대기 시간 및 제한에 대한 메트릭이 캡처되고 보고됩니다.

## 프로그래밍 기능

[+] [테이블 확장](#)

범주	기능
REST (영문)	<p><a href="#">서비스 REST API</a>는 인덱싱, 쿼리 및 AI 보강에 관련된 모든 작업을 포함한 데이터 평면 작업에 사용합니다. 또한 이 클라이언트 라이브러리를 사용하여 시스템 정보와 통계를 검색할 수 있습니다.</p> <p><a href="#">관리 REST API</a>는 서비스를 만들고 Azure Resource Manager를 통해 프로비전하는 데 사용할 수 있습니다. 이 API를 사용하여 키 및 용량을 관리할 수도 있습니다.</p>
Azure SDK for .NET	<p><a href="#">Azure.Search.Documents</a>는 인덱싱, 쿼리 및 AI 보강에 관련된 모든 작업을 포함한 데이터 평면 작업에 사용합니다. 또한 이 클라이언트 라이브러리를 사용하여 시스템 정보와 통계를 검색할 수 있습니다.</p> <p><a href="#">Microsoft.Azure.Management.Search</a>는 서비스를 만들고 Azure Resource Manager를 통해 프로비전하는 데 사용할 수 있습니다. 이 API를 사용하여 키 및 용량을 관리할 수도 있습니다.</p>
Java용 Azure SDK	<p><a href="#">com.azure.search.documents</a>는 인덱싱, 쿼리 및 AI 보강에 관련된 모든 작업을 포함한 데이터 평면 작업에 사용합니다. 또한 이 클라이언트 라이브러리를 사용하여 시스템 정보와 통계를 검색할 수 있습니다.</p> <p><a href="#">com.microsoft.azure.management.search</a>는 서비스를 만들고 Azure Resource Manager를 통해 프로비전하는 데 사용할 수 있습니다. 이 API를 사용하여 키 및 용량을 관리할 수도 있습니다.</p>
Python용 Azure SDK	<p><a href="#">azure-search-documents</a>는 인덱싱, 쿼리 및 AI 보강에 관련된 모든 작업을 포함한 데이터 평면 작업에 사용합니다. 또한 이 클라이언트 라이브러리를 사용하여 시스템 정보와 통계를 검색할 수 있습니다.</p> <p><a href="#">azure-mgmt-search</a>는 서비스를 만들고 Azure Resource Manager를 통해 프로비전하는 데 사용합니다. 이 API를 사용하여 키 및 용량을 관리할 수도 있습니다.</p>
JavaScript/TypeScript용 Azure SDK	<p><a href="#">azure/search-documents</a>는 인덱싱, 쿼리 및 AI 보강에 관련된 모든 작업을 포함한 데이터 평면 작업에 사용합니다. 또한 이 클라이언트 라이브러리를 사용하여 시스템 정보와 통계를 검색할 수 있습니다.</p> <p><a href="#">azure/arm-search</a>는 서비스를 만들고 Azure Resource Manager를 통해 프로비전하는 데 사용합니다. 이 API를 사용하여 키 및 용량을 관리할 수도 있습니다.</p>

## 참고 항목

- [Azure AI 검색의 새로운 기능은 무엇인가요?](#)
- [Azure AI 검색의 미리 보기 기능](#)

# 피드백

이 페이지가 도움이 되었나요?

Yes

No

[제품 사용자 의견 제공](#) | Microsoft Q&A에서 도움말 보기

# 무료로 Azure AI Search 사용해 보기

아티클 • 2024. 10. 18.

Azure를 접하는 경우 무료 평가판 구독을 설정하여 Azure AI Search 및 기타 서비스를 무료로 탐색할 수 있습니다. 사용자 고유의 콘텐츠에 대한 정보 검색은 AI 생성 검색을 비롯한 많은 시나리오에 유용합니다.

이 문서에서는 Azure AI Search 평가를 빠르고 효율적으로 완료할 수 있도록 Azure 평가판 구독에서 가장 많은 가치를 얻는 방법을 설명합니다.

## 1단계: 무료 구독 등록

Azure AI Search를 무료로 사용해 보려면 [평가판 구독](#)을 시작합니다. 평가판 구독은 갱신할 수 없고, 한 달 동안 활성 상태이며, 무료로 청구 가능한 서비스를 만들 수 있도록 무료 크레딧이 제공됩니다.

이 시점에서 크레딧은 USD 200과 동일합니다. 언제나 그처럼 정확한 금액은 변경될 수 있지만 평가판 구독 등록 페이지에서 크레딧을 확인할 수 있습니다.

### 평가판 구독 시작

등록하면 다음 링크 중 하나를 즉시 사용하여 Azure 리소스 및 환경에 액세스할 수 있습니다.

- [Azure Portal](#)에 로그인하여 더 많은 리소스를 보고 관리하고 만듭니다. 포털을 사용하여 크레딧 및 예상 비용을 추적할 수도 있습니다.
- [Azure OPENAI](#)에 모델을 배포하고 정보 검색을 위해 Azure AI Search를 사용하는 코드 없는 접근 방식을 위해 [Azure AI Studio](#)에 로그인합니다. 먼저 여기에서 시작하는 것이 좋습니다.

## 2단계: "1일차" 작업

[Azure AI Studio](#)에서 벡터 인덱스를 빌드하고 사용하는 방법을 시작하는 것이 좋습니다.

- [Azure AI Studio에 로그인합니다](#).
- 새 허브 및 프로젝트를 만듭니다.
- 왼쪽의 구성 요소 아래에서 **인덱스를 선택합니다**. 인덱스 만들기 마법사는 나머지 작업을 안내합니다.

- 원본 데이터 페이지에서 LLM을 사용하여 쿼리하려는 로컬 파일이 있는 경우 업로드합니다.
- 인덱스 설정에서 새 Azure AI Search 서비스를 만들 수 있습니다. 마법사는 일치하는 지역을 자동으로 선택하지만 가격 책정 계층을 선택합니다.

더 큰 데이터 파일 및 더 많은 인덱스에 대한 기본 또는 파일이 50MB 미만인 경우 무료를 사용하는 것이 좋습니다. Basic에는 더 많은 기능과 스토리지가 있지만 서비스 수명 동안 청구할 수 있으며 전체 평가 기간 동안 유지하면 사용 가능한 크레딧의 약 3분의 1을 사용할 수 있습니다.

#### 💡 팁

Azure AI Search 및 Azure OpenAI는 동일한 지역에 있어야 합니다.

## 3단계: 다음 단계에 대한 계획 수립

평가판 기간은 빠르게 진행될 수 있습니다. 작업 계획을 사용하면 평가판 구독을 최대한 활용할 수 있습니다. Azure AI Search의 경우 대부분의 최신 고객과 개발자는 RAG 패턴을 탐색하고 있습니다.

RAG 시나리오의 다음 단계 평가를 위해 다음을 위한 3~5개의 Azure 리소스가 있어야 합니다.

- 데이터 저장
- 포함 및 채팅 모델 배포(Azure OpenAI)
- 인덱싱 중에 AI 생성 콘텐츠를 만들기 위한 AI 서비스 적용(선택 사항)
- 정보 검색 추가(Azure AI Search)
- 프런트 엔드 앱 추가(선택 사항)

대부분의 빠른 시작 및 자습서에서는 Azure Storage를 사용하므로 시작하기 위해 Azure Storage 계정을 만드는 것이 좋습니다.

생성 검색에는 포함 및 채팅 모델이 필요합니다. Azure 클라우드는 Azure OpenAI를 제공 하지만 다중 모드 포함(채팅은 아님)에 Azure AI Vision을 사용할 수도 있습니다. 또 다른 모델 공급자는 Azure AI Studio이며 모델 카탈로그에 채팅 및 포함 모델을 배포합니다. 그러나 초기 탐색의 경우 친숙한 기능과 일반 제품에 대해 Azure OpenAI를 사용하는 것이 좋습니다.

애플리케이션 프런트 엔드는 더 많은 대상을 위한 솔루션을 프로토타입으로 만드는 경우에 유용합니다. Azure Web Apps를 사용하거나 이 작업에 대한 ASP.NET MVC 애플리케이션을 빌드할 수 있습니다. 그렇지 않으면 로컬로 작업하는 경우 Visual Studio Code 또는

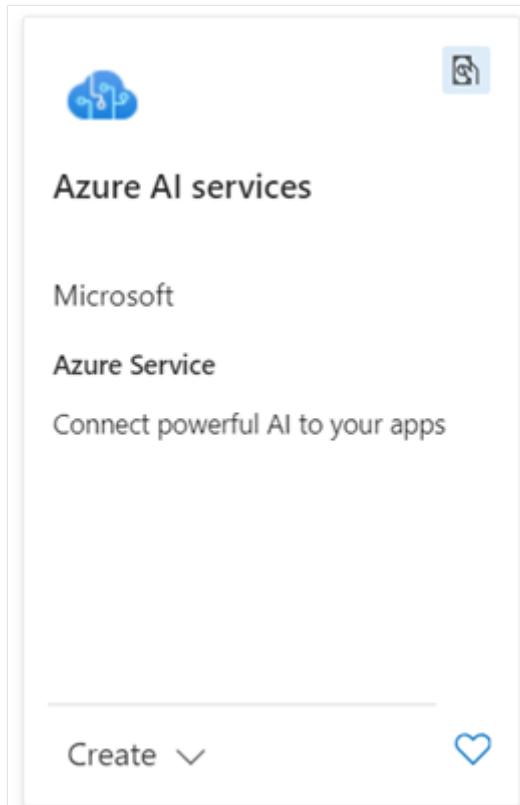
다른 IDE의 Jupyter Notebook에서 출력을 볼 수 있습니다. 또는 localhost에서 실행되는 콘솔 앱 또는 기타 앱에서 결과를 봅니다.

## 서비스 만들기

1. 아직 검색 서비스가 없는 경우 기본 계층과 모델 공급자를 제공하는 지역을 선택하여 검색 서비스를 만듭니다. 대부분의 Azure AI Search 지역은 더 높은 용량의 스토리지 제한을 제공합니다. 이전 및 낮은 제한이 있는 몇 가지만 있습니다. 기본 계층의 경우 설치할 때 15GB 파티션이 있는지 확인합니다.

검색 서비스 만들기

2. Azure Storage 계정을 만들고 범용 계정을 선택하고 기본 설정을 사용합니다.
3. 모델 공급자로 Azure OpenAI 리소스를 만듭니다.
4. 인덱싱 워크로드에 적용된 AI를 사용하고 Azure AI Vision 다중 모드 API를 포함 모델 공급자로 사용하는 Azure AI 다중 서비스 계정을 만듭니다. 적용된 AI를 연결할 수 있는 경우 인덱싱 중에 콘텐츠를 만들고 변환할 수 있습니다. 다중 모드 API의 경우 해당 API를 제공하는 지역을 선택해야 합니다. Azure Marketplace에서 다음 타일을 찾습니다.



## 빠른 시작 시도

Rest 또는 Python 확장과 함께 Visual Studio Code를 사용하는 Azure AI Search 또는 빠른 시작에 대한 포털 빠른 시작을 사용해 보세요. 검색 가능한 콘텐츠를 만드는 가장 빠른 방법이며 작업을 완료하기 위해 코딩 기술이 필요하지 않습니다.

- [빠른 시작: Azure Portal에서 벡터 검색](#)
- [빠른 시작: Azure Portal에서 이미지 검색](#)
- [빠른 시작: Azure Portal의 키워드](#)
- [빠른 시작: Python 클라이언트를 사용한 RAG\(생성 검색\)](#)
- [빠른 시작: REST 클라이언트를 사용하여 벡터 검색](#)

Azure AI Studio 및 Azure OpenAI Studio는 Azure AI Search에서 콘텐츠에 대한 연결을 지원합니다.

- [빠른 시작: Azure OpenAI Studio에서 사용자 고유의 데이터를 사용하여 채팅](#)
- [자습서: 프롬프트 흐름 SDK를 사용하여 사용자 지정 채팅 앱 빌드](#)

개발자는 `azure-search-vector-samples` 리포지토리 또는 [솔루션 가속기를 검토](#) 해야 합니다. Azure 평가판 구독을 사용하여 이러한 샘플을 배포하고 실행할 수 있습니다.

많은 샘플 및 [가속기에](#) 는 모든 Azure 리소스 및 종속성을 배포하는 bicep 스크립트가 함께 제공됩니다. 따라서 설치 단계를 건너뛰고 개발이 완료되는 즉시 운영 솔루션을 탐색 할 수 있습니다.

## 4단계: 크레딧 추적

평가 기간 동안 USD 200 크레딧 할당을 유지하려고 합니다. 대부분의 서비스는 종량제이므로 사용하지 않는 동안에는 요금이 청구되지 않지만 기본 계층의 Azure AI Search 서비스는 전용 클러스터에 프로비전되며 사용자만 사용할 수 있습니다. 수명 동안 청구할 수 있습니다. 기본 계층 검색 서비스를 프로비전하는 경우 Azure AI Search가 평가 기간 동안 사용 가능한 크레딧의 약 3분의 1을 사용할 것으로 예상합니다.

평가 기간 동안 Azure Portal은 오른쪽 위에 사용된 크레딧 수와 남은 크레딧을 알려주는 알림을 제공합니다.

Azure Portal에서 구독을 검색 하여 언제든지 구독 정보를 확인하여 청구를 모니터링할 수도 있습니다. 개요 페이지에서는 지출 비율, 예측 및 비용 관리를 제공합니다. 자세한 내용은 Azure 체험 계정에 [포함된 무료 서비스의 사용량 확인을 참조하세요](#).

## 무료 계층 고려

크레딧을 소비하지 않는 검색 서비스를 만들 수 있습니다. 다음은 유의해야 할 무료 계층에 대한 몇 가지 사항입니다.

- Azure 구독당 하나의 무료 검색 서비스를 사용할 수 있습니다.
- Microsoft Entra ID 인증 및 권한 부여에 대한 의미 체계 순위 및 관리 ID를 특징으로 하는 자습서를 제외하고 모든 빠른 시작 및 대부분의 자습서를 완료할 수 있습니다.
- 스토리지는 50MB로 제한됩니다.
- 한 번에 최대 3개의 인덱스, 인덱서, 데이터 원본 및 기술 세트를 가질 수 있습니다.

무료 계층에 적용되는 다른 제약 조건에 대한 서비스 제한을 검토합니다.

## 다음 단계

Azure 평가판 구독에 등록합니다.

[평가판 구독 시작](#)

준비가 되면 Azure AI Search를 첫 번째 리소스로 추가합니다.

[검색 서비스 만들기](#)

---

## 피드백

이 페이지가 도움이 되었나요?

 Yes

 No

[제품 사용자 의견 제공](#) | [Microsoft Q&A에서 도움말 보기](#)

# Azure AI 검색 질문과 대답

FAQ

Azure AI 검색에 대해 질문과 대답에 대한 답변을 찾아보세요.

## 일반

### Azure AI 검색이란?

Azure AI 검색은 전체 텍스트 및 벡터 검색 시나리오를 위한 검색 가능한 콘텐츠의 전용 검색 엔진 및 영구 스토리지를 제공합니다. 여기에는 원시 콘텐츠에서 더 많은 텍스트와 구조를 추출하고, 벡터 검색을 위해 콘텐츠를 청크화 및 벡터화하는 통합 AI도 포함됩니다.

### Azure AI 검색을 사용하려면 어떻게 해야 하나요?

기본 워크플로는 인덱스 만들기, 로드 및 쿼리입니다. 대부분의 작업에 포털을 사용할 수 있지만 Azure AI 검색은 프로그래밍 방식으로 클라이언트 코드의 요청을 처리하는 데 사용됩니다. 프로그래밍 방식 지원은 Azure용 .NET, Python, Java 및 JavaScript SDK의 REST API 및 클라이언트 라이브러리를 통해 제공됩니다.

### "Azure Search", "Azure Cognitive Search", "Azure AI 검색"은 동일한 제품인가요?

Azure Search는 서비스 작업에서 인식 기술 및 AI 처리의 확장된 사용(아직 선택 사항임)을 반영하기 위해 2019년 10월에 Azure Cognitive Search로 이름이 변경되었습니다.

Azure Cognitive Search는 Azure AI 서비스에 맞게 2023년 10월에 Azure AI 검색으로 이름이 변경되었습니다.

## 어떤 언어가 지원됩니까?

벡터의 경우 사용하는 포함 모델에 따라 언어 환경이 결정됩니다.

비벡터 문자열 및 숫자의 경우 토큰화에 사용되는 기본 분석기는 표준 Lucene이며 언어에 구애받지 않습니다. 그렇지 않으면 언어 지원은 인바운드(인덱스 생성) 및 아웃바운드(검색어) 콘텐츠에 언어 규칙을 적용하는 언어 분석기를 통해 표현됩니다. 맞춤법 검사기 등 일부 기능은 언어의 하위 집합으로 제한됩니다.

## 내 솔루션에 검색을 어떻게 통합하나요?

클라이언트 코드는 Azure SDK 클라이언트 라이브러리 또는 REST API를 호출하여 검색 인덱스에 연결하고, 쿼리를 수식화하고, 응답을 처리해야 합니다. 인덱스를 빌드하고 새로 고치거나 인덱서를 프로그래밍 방식 또는 스크립트로 실행하는 코드를 빌드할 수도 있습니다.

## 다양한 API에 기능적 패리티가 있나요?

항상 그런 것은 아닙니다. 미리 보기 API 버전에서 새 기능을 구현하는 첫 번째 항목은 항상 REST API입니다. Azure SDK의 클라이언트 라이브러리는 시간이 지남에 따라 새로운 기능을 선택하지만 자체 일정에 따라 릴리스됩니다.

REST API가 최신 기능과 함께 가장 먼저 나왔지만 Azure SDK는 더 많은 코딩 지원을 제공합니다. 필요한 기능을 사용할 수 없는 경우가 아니면 REST보다 Azure SDK를 사용하는 것이 좋습니다.

## 서비스를 일시 중지하고 청구를 중지할 수 있나요?

검색 서비스를 일시 중지할 수 없습니다. Azure AI 검색에서는 서비스가 만들어질 때 컴퓨팅 리소스가 할당됩니다. 요청에 따라 이러한 리소스를 해제하고 재청구하는 것은 불가능합니다.

## 서비스를 업그레이드, 다운그레이드, 이름 변경 또는 이동할 수 있나요?

서비스 계층, 이름 및 지역은 서비스 수명 동안 고정됩니다.

## 내 검색 서비스를 다른 구독 또는 리소스 그룹으로 마이그레이션하는 경우 작동 중지 시간이 예상되나요?

[리소스를 이동하기 전에 검사 목록](#)을 따르고 각 단계가 완료되었는지 확인하는 한 중단 시간이 없어야 합니다.

## 인덱싱

## Azure AI 검색에서 "인덱싱"은 무엇을 의미하나요?

검색 인덱스를 채우는 텍스트 콘텐츠 및 토큰의 수집, 구문 분석 및 저장을 말합니다. 인덱싱은 정보 검색을 지원하는 역 인덱스 및 기타 실제 데이터 구조를 만듭니다.

스키마에 벡터 필드가 포함된 경우 벡터 인덱스를 만듭니다.

## 인덱스를 이동, 백업 및 복원할 수 있나요?

인덱스 포팅에 대한 기본 지원은 없습니다. 검색 인덱스는 작동 데이터를 수집하는 다른 데이터 원본의 콘텐츠를 허용하는 다운스트림 데이터 구조로 간주됩니다. 따라서 인덱스를 삭제하거나 이동하려는 경우 원본 데이터에서 인덱스를 다시 빌드해야 하므로 인덱스 백업 및 복원에 대한 기본 제공 지원이 없습니다.

그러나 검색 서비스 간에 인덱스를 이동하려는 경우 이 [Azure AI 검색 .NET 샘플 저장소](#)에서 `index-backup-restore` 샘플 코드를 사용해 볼 수 있습니다. 또한 [Python 버전의 백업 및 복원](#)도 있습니다.

## 인덱스나 서비스가 삭제된 후 복원할 수 있나요?

아니요, Azure AI 검색 인덱스 또는 서비스를 삭제하는 경우 복구할 수 없습니다. 검색 서비스를 삭제하면 서비스의 모든 인덱스가 영구적으로 삭제됩니다.

## SQL Database 복제본에서 인덱싱할 수 있나요?

Azure SQL Database용 검색 인덱서를 사용하는 경우 처음부터 인덱스를 빌드할 때 기본 또는 보조 복제본을 데이터 원본으로 사용하는 데 제한이 없습니다. 그러나 충분 업데이트(변경된 레코드 기준)를 통해 인덱스를 새로 고칠 때는 주 복제본이 필요합니다. 이것은 주 복제본에서만 변경 추적을 보장하는 SQL Database에 따른 요구 사항입니다. 인덱스 새로 고침 워크로드에 보조 복제본을 사용할 경우 전체 데이터를 확보한다고는 보장할 수 없습니다.

## 벡터

### 벡터 검색이란?

벡터 검색은 벡터 표현을 비교하여 가장 유사한 문서를 찾는 기술입니다. 벡터 표현의 목표는 항목의 필수 특성을 숫자 형식으로 캡처하는 것이므로 벡터 쿼리는 키워드 또는 태그를 기반으로 하는 명시적 일치 항목이 없더라도 유사한 콘텐츠를 식별할 수 있습니다. 사용자가 검색을 수행하면 쿼리가 벡터 표현으로 요약되고 벡터 검색 엔진이 가장 유사한 문서를 식별합니다. 대규모 데이터베이스의 효율성을 개선하기 위해 벡터 검색은 쿼리 벡터에 대해 가장 가까운 근사한 인접 항목을 제공하는 경우가 많습니다. Azure AI 검색의 벡터 제품에 대한 자세한 내용은 [벡터 검색 개요](#)를 참조하세요.

## Azure AI 검색은 벡터 검색을 지원하나요?

Azure AI 검색은 벡터 인덱싱 및 검색을 지원합니다. 미리 보기 및 베타 라이브러리를 사용하는 경우 쿼리 문자열 및 콘텐츠를 벡터화할 수 있습니다.

## Azure AI 검색에서 벡터 검색은 어떻게 작동하나요?

독립 실행형 벡터 검색을 사용하면 먼저, 포함 모델을 사용하여 포함 공간 내에서 콘텐츠를 벡터 표현으로 변환합니다. 그런 다음, 인덱싱을 위해 문서 페이로드에서 이러한 벡터를 검색 인덱스에 제공할 수 있습니다. 검색 요청을 제공하려면 인덱싱에서 동일한 DNN(심층 신경망)을 사용하여 검색 쿼리를 벡터 표현으로 변환하고, 벡터 검색은 가장 유사한 벡터를 찾아 해당 문서를 반환합니다.

Azure AI 검색에서는 텍스트 및 기타 유형의 콘텐츠와 함께 벡터 데이터를 문서의 필드로 인덱싱할 수 있습니다. 벡터 필드에는 [여러 데이터 형식](#)이 있습니다.

벡터 쿼리는 독립 실행형 또는 동일한 검색 요청의 용어 쿼리 및 필터를 비롯한 다른 쿼리 형식과 함께 실행할 수 있습니다.

## Azure AI 검색에서 내 콘텐츠 또는 쿼리를 벡터화 할 수 있나요?

[기본 제공 통합 벡터화](#)가 이제 일반 공급합니다.

## 내 검색 서비스에서 벡터 검색을 지원하나요?

대부분의 기존 서비스는 벡터 검색을 지원합니다. 벡터 검색 및 인덱스 생성을 지원하는 패키지 또는 API를 사용하는 경우 기본 검색 서비스가 벡터 검색을 지원하지 않으며 새 서비스를 만들어야 합니다. 이는 2019년 1월 1일 이전에 만든 서비스의 소규모 하위 집합에 대해 발생할 수 있습니다.

## 기존 인덱스로 벡터 검색을 추가할 수 있나요?

검색 서비스에서 벡터 검색을 지원하는 경우 기존 인덱스와 새 인덱스 모두 벡터 필드를 수용할 수 있습니다.

## 새 검색 서비스와 기존 검색 서비스 간에 다른 벡터 인덱스 크기 제한이 표시되는 이유는 무엇인가요?

새로운 검색 서비스에 대해 전 세계적으로 향상된 벡터 인덱스 크기 제한을 둘아웃하고 있지만, 특정 지역에서는 여전히 인프라 용량을 구축하는 중입니다. 지원되는 지역에서 만든 새 검색 서비스에는 증가된 벡터 인덱스 크기 제한이 표시됩니다. 아쉽게도 기존 서비스는 새 제한으로 마이그레이션할 수 없습니다.

## 검색 인덱스에서 벡터 검색을 사용하도록 설정하려면 어떻게 해야 하나요?

인덱스에서 벡터 검색을 사용하도록 설정하려면 다음을 수행해야 합니다.

- 필드 컬렉션에 하나 이상의 벡터 필드를 추가합니다.
- HNSW와 등, 사용된 근사값 인접 알고리즘의 매개 변수를 포함하여 벡터 검색 필드에서 사용되는 구성을 지정하는 인덱스 스키마에 "vectorSearch" 섹션을 추가합니다.
- 최신 안정 버전[2024-07-01](#) 또는 Azure SDK를 사용하여 인덱스를 만들거나 업데이트하고, 문서를 로드하고, 쿼리를 실행합니다.

## 쿼리

### 쿼리 실행은 어디에서 발생하나요?

쿼리는 검색 서비스에서 호스트되는 단일 검색 인덱스를 통해 실행됩니다. 여러 인덱스를 조인하여 둘 이상의 인덱스에서 콘텐츠를 쿼리할 수는 없지만 [여러 쿼리 서비스에서 동일한 이름의 인덱스를 쿼리](#) 할 수 있습니다.

### 유효하다고 알고 있는 용어에 대해 일치가 0인 이유는 무엇인가요?

가장 일반적인 경우는 각 쿼리 입력이 언어적 분석의 다른 검색 동작과 수준을 지원하는 것을 인지하지 못하기 때문입니다. 핵심 작워크로드인 전체 텍스트 검색에는 용어를 근본 형태로 분석하는 언어 분석 단계가 포함됩니다. 토큰화된 용어는 더 많은 변형 항목과 일치하기 때문에 이 쿼리 구문 분석 단계에는 가능한 일치보다 범위가 더 넓어집니다.

그러나 와일드카드, 유사 항목 및 regex 쿼리는 일반적인 용어 또는 구문 쿼리처럼 분석되지 않으며, 쿼리가 검색 인덱스에 있는 분석된 형태의 단어와 일치하지 않을 경우 제대로 불러올 수 없습니다. 쿼리 구문 분석 및 분석에 대한 자세한 내용은 [쿼리 아키텍처](#)를 참조하세요.

## 내 와일드카드 검색이 느린 이유는 무엇인가요?

접두사, 유사 항목 및 regex와 같은 대부분의 와일드카드 검색 쿼리는 검색 인덱스에서 일치하는 용어를 사용하여 내부적으로 다시 작성됩니다. 이 추가 처리로 대기 시간이 늘어납니다. 나아가, 많은 용어로 다시 작성될 가능성이 큰 `a*`와 같은 광범위한 검색 쿼리는 느릴 수 있습니다. 고성능 와일드카드 검색을 수행하려면 [사용자 지정 분석기](#)를 정의하는 것이 좋습니다.

## 여러 인덱스에서 검색할 수 있나요?

아니요, 쿼리는 항상 단일 인덱스로 범위가 지정됩니다.

## 모든 경기에서 검색 점수가 1.0으로 일정하게 유지되는 이유는 무엇인가요?

전체 텍스트 검색어에 대한 검색 점수는 [일치하는 용어의 통계적 속성](#)을 기반으로 생성되며 결과 집합에서 높은 순서로 정렬됩니다. 전체 텍스트 쿼리가 아닌 쿼리 형식(와일드카드, 접두사, 정규식)은 관련성 점수로 순위가 매겨지지 않습니다. 이 동작은 의도된 것입니다. 상수 점수를 통해 쿼리 확장을 통해 찾은 일치 항목을 순위에 영향을 주지 않고 결과에 포함할 수 있습니다.

예를 들어, 와일드카드 검색에서 "tour\*"를 입력하면 "tours", "tourettes" 및 "tourmaline"이 일치합니다. 이러한 결과의 특성을 감안할 때, 어떤 용어가 더 중요한지 합리적으로 유추할 방법이 없습니다. 이러한 이유로 와일드카드, 접두사 및 정규식 형식의 쿼리에서 결과 점수를 매길 때 용어 빈도는 무시됩니다. 예기치 않은 일치의 가능성에 따른 왜곡을 방지하기 위해 부분 이력 기준 검색 결과에는 일정 점수가 부여됩니다.

## 보안

## Azure AI 검색은 고객 데이터를 어디에 저장하나요?

서비스가 배포되는 [지역\(Geo\)](#)에 데이터를 저장합니다. Microsoft는 고가용성과 내구성을 위해 동일한 지역 내에서 데이터를 복제할 수 있습니다. 자세한 내용은 [Azure의 데이터 보존](#)을 참조하세요.

## Azure AI 검색은 처리를 위해 고객 데이터를 다른 서비스로 전송하나요?

예, 기술 및 벡터라이저는 포함 또는 채팅을 위해 지정한 다른 Azure 리소스 또는 외부 모델에 대해 Azure AI 검색에서 아웃바운드 호출을 수행합니다. 이러한 API에 대한 호출에는 일반적으로 처리할 원시 콘텐츠나 포함 모델에 의해 벡터화되는 쿼리가 포함됩니다. Azure 간 연결의 경우 서비스는 내부 네트워크를 통해 요청을 보냅니다. 사용자 지정 기술이나 벡터라이저를 추가하는 경우 공유 프라이빗 링크를 구성하지 않는 한 인덱서는 공용 네트워크를 통해 사용자 지정 기술에 제공된 URI로 콘텐츠를 보냅니다.

## Azure AI 검색은 다른 지역의 고객 데이터를 처리하나요?

기술에 사용되는 Azure AI 서비스를 호스팅하는 지역, 사용자 지정 기술을 호스팅하는 Azure 앱이나 함수 또는 배포된 모델을 호스팅하는 Azure OpenAI나 Azure AI 스튜디오 지역에서 처리(벡터화 또는 적용된 AI 변환)가 수행됩니다. 이러한 리소스는 사용자가 지정하므로 검색 서비스와 동일한 지역에서 프로비전할지 여부를 선택할 수 있음

외부(비 Azure) 모델이나 서비스로 데이터를 보내는 경우 처리 위치는 외부 서비스에 의해 결정됩니다.

## 사용자 ID를 기반으로 검색 결과에 대한 액세스를 제어할 수 있나요?

문서를 사용자 ID와 연결하는 솔루션을 구현하면 가능합니다. 일반적으로 애플리케이션을 실행할 수 있는 권한이 있는 사용자는 모든 검색 결과를 볼 수 있는 권한도 있습니다. Azure AI 검색은 행 수준 또는 문서 수준 권한을 기본적으로 지원하지 않지만 해결 방법으로 보안 필터를 구현할 수 있습니다. 단계와 스크립트는 RAG를 사용하여 Python enterprise 채팅 샘플 시작하기를 참조하세요.

## 사용자 ID를 기반으로 작업에 대한 액세스를 제어할 수 있나요?

예, 콘텐츠에 대한 데이터 평면 작업에 역할 기반 권한 부여를 사용할 수 있습니다.

## 검색 서비스가 IP 방화벽이나 프라이빗 엔드포인트 뒤에 있는 경우 Azure Portal을 사용하여 검색 콘텐츠를 보고 관리할 수 있나요?

클라이언트 및 포털 액세스를 허용하는 네트워크 예외를 만드는 경우 네트워크 보호 검색 서비스에서 Azure Portal을 사용할 수 있습니다. 자세한 내용은 IP 방화벽을 통해 연결 또는 프라이빗 엔드포인트를 통해 연결을 참조하세요.

# 다음 단계

질문에 대한 대답이 여기에 없으면 다음 출처에서 추가 질문 및 대답을 참조할 수 있습니다.

[스택 오버플로: Azure AI 검색 ↗](#)

[Azure AI 검색에서 전체 텍스트 검색이 작동하는 방식](#)

[Azure AI 검색이란?](#)

---

## 피드백

이 페이지가 도움이 되었나요?

 Yes

 No

[제품 사용자 의견 제공 ↗](#) | [Microsoft Q&A에서 도움말 보기](#)

# 빠른 시작: REST를 사용한 벡터 검색

아티클 • 2024. 10. 31.

검색 REST API를 사용하여 Azure AI 검색에서 벡터를 만들고, 로드하고, 쿼리하는 방법을 알아봅니다.

Azure AI 검색에서 [벡터 저장소](#)에는 벡터 필드와 비 벡터 필드를 정의하는 인덱스 스키마, 포함 공간을 만드는 알고리즘에 대한 벡터 구성, 쿼리 요청에 사용되는 벡터 필드 정의에 대한 설정이 있습니다. [인덱스 만들기](#) API는 벡터 저장소를 만듭니다.

Azure 구독이 아직 없는 경우 시작하기 전에 [체험 계정](#)을 만듭니다.

## ① 참고

이 빠른 시작에서는 벡터화 단계를 생략하고 샘플 문서에 포함을 제공합니다. 자체 콘텐츠에 대해 [기본 제공 데이터 청크화 및 벡터화](#)를 추가하려면 [데이터 가져오기 및 벡터화 마법사](#)를 사용하여 엔드투엔드 연습을 시도해 보세요.

## 필수 조건

- Visual Studio Code 와 REST 클라이언트. 시작에 도움이 필요한 경우 [빠른 시작: REST를 사용한 텍스트 검색](#)을 참조하세요.
- Azure AI 검색(지역 및 계층 무관). 이 빠른 시작에서는 무료 계층을 사용할 수 있지만 더 큰 데이터 파일에는 기본 이상을 사용하는 것이 좋습니다. 현재 구독에서 [기존 Azure AI 검색 리소스를 만들거나 찾습니다](#).

대부분의 기존 서비스는 벡터 검색을 지원합니다. 2019년 1월 이전에 만들어진 서비스의 작은 하위 집합의 경우 벡터 필드가 포함된 인덱스는 생성 시 실패합니다. 이런 상황에서는 새로운 서비스를 만들어야 합니다.

- (선택 사항) 의미 체계 순위 재지정을 호출하는 쿼리 예제를 실행하려면 검색 서비스가 기본 계층 이상이어야 하며 [의미 순위 매기기가 활성화](#)되어 있어야 합니다.
- (선택 사항) `text-embedding-ada-002` 가 배포된 Azure OpenAI 리소스. 원본 `.rest` 파일에는 새 텍스트 포함을 생성하기 위한 선택적 단계가 포함되어 있지만 사전 생성된 포함이 제공되므로 이 종속성을 생략할 수 있습니다.

## 파일 다운로드

GitHub에서 REST 샘플을 다운로드 [하여](#) 이 빠른 시작에서 요청을 보냅니다. 자세한 내용은 [GitHub에서 파일 다운로드](#)를 참조하세요.

이 문서의 지침을 사용하여 로컬 시스템에서 새 파일을 시작하고 수동으로 요청을 만들 수도 있습니다.

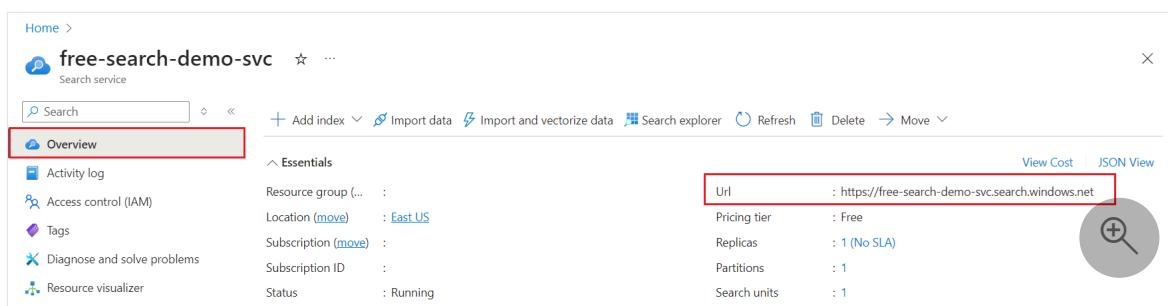
## 검색 서비스 엔드포인트 가져오기

Azure Portal에서 검색 서비스 엔드포인트를 찾을 수 있습니다.

1. [Azure Portal](#)에 로그인하고 [검색 서비스를 찾습니다](#).

2. **개요** 홈페이지에서 URL을 찾습니다. 엔드포인트의 예는 다음과 같습니다.

`https://mydemo.search.windows.net`



The screenshot shows the Azure Portal interface for a search service named 'free-search-demo-svc'. The 'Overview' tab is selected, indicated by a red box. On the right, there's a table of service details. One row in the table, 'Url', is also highlighted with a red box. The table data is as follows:

Essentials	
Resource group (move)	: East US
Subscription (move)	:
Subscription ID	:
Status	: Running
Url	: <a href="https://free-search-demo-svc.search.windows.net">https://free-search-demo-svc.search.windows.net</a>
Pricing tier	: Free
Replicas	: 1 (No SLA)
Partitions	: 1
Search units	: 1

이후 단계에서 이 엔드포인트를 `.rest` 또는 `.http` 파일에 붙여넣습니다.

## 액세스 구성

검색 엔드포인트에 대한 요청은 인증 및 권한 부여를 받아야 합니다. 이 작업에 API 키 또는 역할을 사용할 수 있습니다. 키는 시작하기가 더 쉽지만 역할이 더 안전합니다.

역할 기반 연결의 경우 다음 지침에 따라 클라이언트 앱의 ID가 아닌 사용자 ID로 Azure AI 검색에 연결합니다.

### 옵션 1: 키 사용

설정>키를 선택한 다음, 관리자 키를 복사합니다. 관리자 키는 개체를 추가, 수정, 삭제하는 데 사용됩니다. 교환 가능한 관리자 키는 2개입니다. 둘 중 하나를 복사합니다. 자세한 내용은 [키 인증을 사용하여 Azure AI 검색에 연결](#)을 참조하세요.

The screenshot shows the Azure Portal interface for managing a search service. The top navigation bar includes 'Home > free-search-demo-svc'. Below it, there's a key icon and the service name 'free-search-demo-svc | Keys'. On the left, a sidebar lists various service management options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource visualizer, Search management, Settings, Semantic ranker, Knowledge Center, and Keys. The 'Keys' option is highlighted with a red box. The main content area is titled 'API Access control' and contains sections for 'API keys' (selected), 'Role-based access control', and 'Both'. Under 'Manage admin keys', there are fields for 'Primary admin key' and 'Secondary admin key', each with a 'Regenerate' button. A large circular button with a plus sign and magnifying glass is visible on the right.

이후 단계에서 이 키를 `.rest` 또는 `.http` 파일에 붙여넣습니다.

## 옵션 2: 역할 사용

검색 서비스가 [역할 기반 액세스에 대해 구성](#)되어 있는지 확인합니다. [개발자 액세스를 위한 역할 할당](#)이 미리 구성되어 있어야 합니다. 역할 할당은 검색 인덱스를 생성, 로드 및 쿼리할 수 있는 권한을 부여해야 합니다.

이 섹션에서는 Azure CLI, Azure PowerShell 또는 Azure Portal을 사용하여 개인 ID 토큰을 가져옵니다.

The first screenshot shows the Azure CLI interface with the command 'az login' entered. The second screenshot shows the Azure CLI interface with the command 'az account get-access-token --scope https://search.azure.com/.default' entered.

이후 단계에서 개인 ID 토큰을 `.rest` 또는 `.http` 파일에 붙여넣습니다.

① 참고

이 섹션에서는 사용자를 대신하여 Azure AI 검색에 연결하는 로컬 클라이언트를 사용한다고 가정합니다. 애플리케이션이 Microsoft Entra ID에 등록되어 있다고 가정하고 [클라이언트 앱에 대한 토큰을 가져오는](#) 방식을 사용할 수도 있습니다.

## 벡터 인덱스 만들기

[인덱스 만들기\(REST\)](#)는 벡터 인덱스를 만들고 검색 서비스에 물리적 데이터 구조를 설정합니다.

인덱스 스키마는 호텔 콘텐츠를 중심으로 구성됩니다. 샘플 데이터는 벡터 및 비벡터 이름과 7개의 가상 호텔에 대한 설명으로 구성됩니다. 이 스키마에는 벡터 인덱싱 및 쿼리, 의미 체계 순위에 대한 구성이 포함됩니다.

1. Visual Studio Code에서 새 텍스트 파일을 엽니다.
2. 변수를 이전에 수집한 값으로 설정합니다. 이 예에서는 개인 ID 토큰을 사용합니다.

```
HTTP

@baseUrl = PUT-YOUR-SEARCH-SERVICE-URL-HERE
@token = PUT-YOUR-PERSONAL-IDENTITY-TOKEN-HERE
```

3. 파일을 `.rest` 또는 `.http` 파일 확장명을 사용하여 저장합니다.
4. 다음 예제를 붙여넣어 검색 서비스에 `hotels-vector-quickstart` 인덱스를 만듭니다.

```
HTTP

### Create a new index
POST {{baseUrl}}/indexes?api-version=2023-11-01 HTTP/1.1
Content-Type: application/json
Authorization: Bearer {{token}}


{
    "name": "hotels-vector-quickstart",
    "fields": [
        {
            "name": "HotelId",
            "type": "Edm.String",
            "searchable": false,
            "filterable": true,
            "retrievable": true,
            "sortable": false,
            "facetable": false,
            "key": true
        },
        {
            "name": "Name",
            "type": "Edm.String",
            "searchable": true,
            "filterable": true,
            "retrievable": true,
            "sortable": true,
            "facetable": true,
            "key": false
        },
        {
            "name": "Address",
            "type": "Edm.String",
            "searchable": true,
            "filterable": true,
            "retrievable": true,
            "sortable": true,
            "facetable": true,
            "key": false
        },
        {
            "name": "Description",
            "type": "Edm.String",
            "searchable": true,
            "filterable": true,
            "retrievable": true,
            "sortable": true,
            "facetable": true,
            "key": false
        },
        {
            "name": "Rating",
            "type": "Edm.Int32",
            "searchable": true,
            "filterable": true,
            "retrievable": true,
            "sortable": true,
            "facetable": true,
            "key": false
        },
        {
            "name": "ReviewsCount",
            "type": "Edm.Int32",
            "searchable": true,
            "filterable": true,
            "retrievable": true,
            "sortable": true,
            "facetable": true,
            "key": false
        }
    ]
}
```

```
{  
    "name": "HotelName",  
    "type": "Edm.String",  
    "searchable": true,  
    "filterable": false,  
    "retrievable": true,  
    "sortable": true,  
    "facetable": false  
},  
{  
    "name": "HotelNameVector",  
    "type": "Collection(Edm.Single)",  
    "searchable": true,  
    "retrievable": true,  
    "dimensions": 1536,  
    "vectorSearchProfile": "my-vector-profile"  
},  
{  
    "name": "Description",  
    "type": "Edm.String",  
    "searchable": true,  
    "filterable": false,  
    "retrievable": true,  
    "sortable": false,  
    "facetable": false  
},  
{  
    "name": "DescriptionVector",  
    "type": "Collection(Edm.Single)",  
    "searchable": true,  
    "retrievable": true,  
    "dimensions": 1536,  
    "vectorSearchProfile": "my-vector-profile"  
},  
{  
    "name": "Category",  
    "type": "Edm.String",  
    "searchable": true,  
    "filterable": true,  
    "retrievable": true,  
    "sortable": true,  
    "facetable": true  
},  
{  
    "name": "Tags",  
    "type": "Collection(Edm.String)",  
    "searchable": true,  
    "filterable": true,  
    "retrievable": true,  
    "sortable": false,  
    "facetable": true  
},  
{  
    "name": "Address",  
    "type": "Edm.ComplexType",
```

```
        "fields": [
            {
                "name": "City", "type": "Edm.String",
                "searchable": true, "filterable": true,
                "retrievable": true, "sortable": true, "facetable": true
            },
            {
                "name": "StateProvince", "type": "Edm.String",
                "searchable": true, "filterable": true,
                "retrievable": true, "sortable": true, "facetable": true
            }
        ],
        {
            "name": "Location",
            "type": "Edm.GeographyPoint",
            "searchable": false,
            "filterable": true,
            "retrievable": true,
            "sortable": true,
            "facetable": false
        }
    ],
    "vectorSearch": {
        "algorithms": [
            {
                "name": "my-hnsw-vector-config-1",
                "kind": "hnsw",
                "hnswParameters":
                {
                    "m": 4,
                    "efConstruction": 400,
                    "efSearch": 500,
                    "metric": "cosine"
                }
            },
            {
                "name": "my-hnsw-vector-config-2",
                "kind": "hnsw",
                "hnswParameters":
                {
                    "m": 4,
                    "metric": "euclidean"
                }
            },
            {
                "name": "my-eknn-vector-config",
                "kind": "exhaustiveKnn",
                "exhaustiveKnnParameters":
                {
                    "metric": "cosine"
                }
            }
        ],
        "profiles": [
```

```

    {
        "name": "my-vector-profile",
        "algorithm": "my-hnsw-vector-config-1"
    }
]
},
"semantic": {
    "configurations": [
        {
            "name": "my-semantic-config",
            "prioritizedFields": {
                "titleField": {
                    "fieldName": "HotelName"
                },
                "prioritizedContentFields": [
                    { "fieldName": "Description" }
                ],
                "prioritizedKeywordsFields": [
                    { "fieldName": "Tags" }
                ]
            }
        }
    ]
}
}

```

## 5. 요청 보내기를 선택합니다. 요청을 보내려면 REST 클라이언트가 필요합니다.

`HTTP/1.1 201 Created` 응답이 있어야 합니다. 응답 본문에는 인덱스 스키마의 JSON 표현이 포함되어어야 합니다.

주요 정보:

- `fields` 컬렉션에는 텍스트 및 벡터 검색에 필요한 키 필드, 텍스트 및 벡터 필드(예: `Description` 및 `DescriptionVector`)가 포함되어 있습니다. 동일한 인덱스에 벡터 필드와 벡터가 아닌 필드를 같은 위치에 배치하면 하이브리드 쿼리가 가능해집니다. 예를 들어 필터, 의미 체계 순위를 사용하는 텍스트 검색, 벡터를 단일 쿼리 작업으로 결합할 수 있습니다.
- 벡터 필드는 `dimensions` 및 `vectorSearchProfile` 속성이 있는 `type: Collection(Edm.Single)` 이어야 합니다.
- `vectorSearch` 섹션은 근사 가장 인접한 항목 알고리즘 구성 및 프로필의 배열입니다. 지원되는 알고리즘에는 HNSW(Hierarchical Navigable Small World)와 철저한 k-가장 인접한 항목이 포함됩니다. 자세한 내용은 [벡터 검색의 관련성 점수](#)를 참조하세요.
- [선택 사항]: `semantic` 구성을 사용하면 검색 결과의 순위 지정을 다시 지정할 수 있습니다. 구성에 지정된 문자열 필드에 대한 `semantic` 형식의 쿼리에서 결과 순위를 다시 지정할 수 있습니다. 자세한 내용은 [의미 체계 순위 지정 개요](#)를 참조하세요.

# 문서 업로드

인덱스 만들기 및 로드는 별도의 단계입니다. Azure AI 검색에서 인덱스에는 검색 가능한 모든 데이터가 포함되어 있으며 검색 서비스에서 실행되는 쿼리가 있습니다. REST 호출의 경우 데이터는 JSON 문서로 제공됩니다. 이 작업에는 [문서- 인덱스 REST API](#)를 사용합니다.

`docs` 컬렉션 및 `index` 작업을 포함하도록 URI가 길어집니다.

## ① 중요

다음 예는 실행할 수 없는 코드입니다. 각각 1,536개의 포함이 있으므로 이 문서에 추가하기에는 너무 길어 가독성을 위해 벡트 값을 제외했습니다. 이 단계를 시도하려면 [GitHub의 샘플](#)에서 실행 가능한 코드를 복사하세요.

HTTP

```
### Upload documents
POST {{baseUrl}}/indexes/hotels-quickstart-vectors/docs/index?api-
version=2023-11-01 HTTP/1.1
Content-Type: application/json
Authorization: Bearer {{token}}


{
  "value": [
    {
      "@search.action": "mergeOrUpload",
      "HotelId": "1",
      "HotelName": "Stay-Kay City Hotel",
      "HotelNameVector": [VECTOR ARRAY OMITTED],
      "Description":
        "The hotel is ideally located on the main commercial artery
of the city
          in the heart of New York.",
      "DescriptionVector": [VECTOR ARRAY OMITTED],
      "Category": "Boutique",
      "Tags": [
        "pool",
        "air conditioning",
        "concierge"
      ],
    },
    {
      "@search.action": "mergeOrUpload",
      "HotelId": "2",
      "HotelName": "Old Century Hotel",
      "HotelNameVector": [VECTOR ARRAY OMITTED],
      "Description":
        "The hotel is situated in a nineteenth century plaza, which
```

has been expanded and renovated to the highest architectural standards to create a modern, functional and first-class hotel in which art and unique historical elements coexist with the most modern comforts.",  
"DescriptionVector": [VECTOR ARRAY OMITTED],  
"Category": "Boutique",  
"Tags": [  
    "pool",  
    "air conditioning",  
    "free wifi",  
    "concierge"  
]  
,  
{  
    "@search.action": "mergeOrUpload",  
    "HotelId": "3",  
    "HotelName": "Gastronomic Landscape Hotel",  
    "HotelNameVector": [VECTOR ARRAY OMITTED],  
    "Description":  
        "The Hotel stands out **for** its gastronomic excellence under the management of William Dough, who advises on and oversees all of the Hotel's restaurant services.",  
        "DescriptionVector": [VECTOR ARRAY OMITTED],  
        "Category": "Resort and Spa",  
        "Tags": [  
            "air conditioning",  
            "bar",  
            "continental breakfast"  
]  
}  
{  
    "@search.action": "mergeOrUpload",  
    "HotelId": "4",  
    "HotelName": "Sublime Palace Hotel",  
    "HotelNameVector": [VECTOR ARRAY OMITTED],  
    "Description":  
        "Sublime Palace Hotel is located **in** the heart of the historic center of Sublime in an extremely vibrant and lively area within short walking distance to the sites and landmarks of the city and is surrounded by the extraordinary beauty of churches, buildings, shops and monuments.  
        Sublime Palace is part of a lovingly restored 1800 palace.",  
        "DescriptionVector": [VECTOR ARRAY OMITTED],  
        "Category": "Boutique",  
        "Tags": [  
            "concierge",  
            "view",  
            "24-hour front desk service"  
]  
},

```
{  
    "@search.action": "mergeOrUpload",  
    "HotelId": "13",  
    "HotelName": "Luxury Lion Resort",  
    "HotelNameVector": [VECTOR ARRAY OMITTED],  
    "Description":  
        "Unmatched Luxury. Visit our downtown hotel to indulge in  
luxury  
            accommodations. Moments from the stadium, we feature the  
best in comfort",  
        "DescriptionVector": [VECTOR ARRAY OMITTED],  
        "Category": "Resort and Spa",  
        "Tags": [  
            "view",  
            "free wifi",  
            "pool"  
        ]  
    },  
    {  
        "@search.action": "mergeOrUpload",  
        "HotelId": "48",  
        "HotelName": "Nordick's Valley Motel",  
        "HotelNameVector": [VECTOR ARRAY OMITTED],  
        "Description":  
            "Only 90 miles (about 2 hours) from the nation's capital and  
nearby  
                most everything the historic valley has to offer. Hiking?  
Wine Tasting? Exploring  
                    the caverns? It's all nearby and we have specially priced  
packages to help make  
                our B&B your home base for fun while visiting the valley.",  
        "DescriptionVector": [VECTOR ARRAY OMITTED],  
        "Category": "Boutique",  
        "Tags": [  
            "continental breakfast",  
            "air conditioning",  
            "free wifi"  
        ],  
    },  
    {  
        "@search.action": "mergeOrUpload",  
        "HotelId": "49",  
        "HotelName": "Swirling Currents Hotel",  
        "HotelNameVector": [VECTOR ARRAY OMITTED],  
        "Description":  
            "Spacious rooms, glamorous suites and residences, rooftop  
pool, walking  
                access to shopping, dining, entertainment and the city  
center.",  
        "DescriptionVector": [VECTOR ARRAY OMITTED],  
        "Category": "Luxury",  
        "Tags": [  
            "air conditioning",  
            "laundry service",  
            "24-hour front desk service"  
        ]  
    }  
}
```

```
        ]  
    }  
}
```

주요 정보:

- 페이로드의 문서는 인덱스 스키마에 정의된 필드로 구성됩니다.
- 벡터 필드에는 부동 소수점 값이 포함되어 있습니다. 차원 특성에는 각각 최소 2개, 최대 3,072개의 부동 소수점 값이 있습니다. 이 빠른 시작에서는 차원 특성을 1,536으로 설정합니다. 이 값이 Azure OpenAI의 **text-embedding-ada-002** 모델에서 생성된 포함 크기이기 때문입니다.

## 쿼리 실행

이제 문서가 로드되었으므로 [문서 - 게시물 검색\(REST\)](#)을 사용하여 문서에 대한 벡터 쿼리를 실행할 수 있습니다.

다양한 패턴을 보여 주는 몇 가지 쿼리가 있습니다.

- 단일 벡터 검색
- 필터를 사용한 단일 벡터 검색
- 하이브리드 검색
- 필터를 사용한 의미 체계 하이브리드 검색

이 섹션의 벡터 쿼리는 다음 두 문자열을 기반으로 합니다.

- **검색 문자열:** `historic hotel walk to restaurants and shopping`
- **벡터 쿼리 문자열(수학적 표현으로 벡터화):** `classic lodging near running trails, eateries, retail`

벡터 쿼리 문자열은 검색 문자열과 의미 체계상 유사하지만 검색 인덱스에 존재하지 않는 용어를 포함합니다. `classic lodging near running trails, eateries, retail`에 대한 키워드 검색을 수행하면 결과는 0입니다. 이 예를 사용하여 일치하는 용어가 없더라도 관련 결과를 가져올 수 있는 방법을 보여 줍니다.

### ⓘ 중요

다음 예는 실행할 수 없는 코드입니다. 각 배열에 1,536개의 포함이 있으므로 이 문서에 추가하기에는 너무 길어 가독성을 위해 벡트 값을 제외했습니다. 이 쿼리를 사용해 보려면 [GitHub의 샘플](#)에서 실행 가능한 코드를 복사하세요.

# 단일 벡터 검색

1. POST 요청에 붙여넣어 검색 인덱스를 쿼리합니다. 그런 다음, **요청 보내기**를 선택합니다. `/docs/search` 연산자를 포함하도록 URI가 길어집니다.

HTTP

```
### Run a query
POST {{baseUrl}}/indexes/hotels-vector-quickstart/docs/search?api-
version=2023-11-01 HTTP/1.1
Content-Type: application/json
Authorization: Bearer {{token}}


{
  "count": true,
  "select": "HotelId, HotelName, Description, Category",
  "vectorQueries": [
    {
      "vector": [0.01944167, 0.0040178085
                . . . TRIMMED FOR BREVITY
                010858015, -0.017496133],
      "k": 7,
      "fields": "DescriptionVector",
      "kind": "vector",
      "exhaustive": true
    }
  ]
}
```

이 벡터 쿼리는 간결성을 위해 단축되었습니다. `vectorQueries.vector`는 쿼리 입력의 벡터화된 텍스트를 포함하고, `fields`는 검색할 벡터 필드를 결정하고, `k`는 반환 할 가장 인접한 항목의 수를 지정합니다.

벡터 쿼리 문자열은 `classic lodging near running trails, eateries, retail`이며, 이 쿼리에서 1,536개의 포함으로 벡터화됩니다.

2. 응답을 검토합니다. `classic lodging near running trails, eateries, retail`에 해당하는 벡터에 대한 응답에는 7개의 결과가 포함됩니다. 각 결과는 검색 점수와 `select`에 나열된 필드를 제공합니다. 유사성 검색에서 응답에는 항상 유사성 점수 값을 기준으로 순서가 지정된 `k`개의 결과가 포함됩니다.

JSON

```
{
  "@odata.context": "https://my-demo-
search.search.windows.net/indexes('hotels-vector-
quickstart')/$metadata#docs(*)",
  "@odata.count": 7,
```

```
"value": [
  {
    "@search.score": 0.857736,
    "HotelName": "Nordick's Valley Motel",
    "Description": "Only 90 miles (about 2 hours) from the nation's capital and nearby most everything the historic valley has to offer. Hiking? Wine Tasting? Exploring the caverns? It's all nearby and we have specially priced packages to help make our B&B your home base for fun while visiting the valley."
  },
  {
    "@search.score": 0.8399129,
    "HotelName": "Swirling Currents Hotel",
    "Description": "Spacious rooms, glamorous suites and residences, rooftop pool, walking access to shopping, dining, entertainment and the city center."
  },
  {
    "@search.score": 0.8383954,
    "HotelName": "Luxury Lion Resort",
    "Description": "Unmatched Luxury. Visit our downtown hotel to indulge in luxury accommodations. Moments from the stadium, we feature the best in comfort"
  },
  {
    "@search.score": 0.8254346,
    "HotelName": "Sublime Palace Hotel",
    "Description": "Sublime Palace Hotel is located in the heart of the historic center of Sublime in an extremely vibrant and lively area within short walking distance to the sites and landmarks of the city and is surrounded by the extraordinary beauty of churches, buildings, shops and monuments. Sublime Palace is part of a lovingly restored 1800 palace."
  },
  {
    "@search.score": 0.82380056,
    "HotelName": "Stay-Kay City Hotel",
    "Description": "The hotel is ideally located on the main commercial artery of the city in the heart of New York."
  },
  {
    "@search.score": 0.81514084,
    "HotelName": "Old Century Hotel",
    "Description": "The hotel is situated in a nineteenth century plaza, which has been expanded and renovated to the highest architectural standards to create a modern, functional and first-class hotel in which art and unique historical elements coexist with the most modern comforts."
  },
  {
    "@search.score": 0.8133763,
    "HotelName": "Gastronomic Landscape Hotel",
    "Description": "The Hotel stands out for its gastronomic excellence under the management of William Dough, who advises on and oversees all of the Hotel's restaurant services."
  }
]
```

```
        }
    ]
}
```

## 필터를 사용한 단일 벡터 검색

필터를 추가할 수 있지만 필터는 인덱스의 벡터가 아닌 콘텐츠에 적용됩니다. 이 예에서 필터가 `Tags` 필드에 적용되어 무료 Wi-Fi를 제공하지 않는 호텔을 필터링합니다.

1. POST 요청에 붙여넣어 검색 인덱스를 쿼리합니다.

HTTP

```
### Run a vector query with a filter
POST {{baseUrl}}/indexes/hotels-vector-quickstart/docs/search?api-
version=2023-11-01 HTTP/1.1
Content-Type: application/json
Authorization: Bearer {{token}}


{
  "count": true,
  "select": "HotelId, HotelName, Category, Tags, Description",
  "filter": "Tags/any(tag: tag eq 'free wifi')",
  "vectorFilterMode": "postFilter",
  "vectorQueries": [
    {
      "vector": [ VECTOR OMITTED ],
      "k": 7,
      "fields": "DescriptionVector",
      "kind": "vector",
      "exhaustive": true
    },
  ]
}
```

2. 응답을 검토합니다. 쿼리는 이전 예와 동일하지만 후처리 제외 필터를 포함하고 있으며 무료 Wi-Fi가 있는 호텔 세 곳만 반환합니다.

JSON

```
{
  "@odata.count": 3,
  "value": [
    {
      "@search.score": 0.857736,
      "HotelName": "Nordick's Valley Motel",
      "Description": "Only 90 miles (about 2 hours) from the
nation's capital and nearby most everything the historic valley has to
```

```

offer. Hiking? Wine Tasting? Exploring the caverns? It's all nearby
and we have specially priced packages to help make our B&B your home
base for fun while visiting the valley.",

    "Tags": [
        "continental breakfast",
        "air conditioning",
        "free wifi"
    ],
},
{
    "@search.score": 0.8383954,
    "HotelName": "Luxury Lion Resort",
    "Description": "Unmatched Luxury. Visit our downtown hotel
to indulge in luxury accommodations. Moments from the stadium, we
feature the best in comfort",
    "Tags": [
        "view",
        "free wifi",
        "pool"
    ],
},
{
    "@search.score": 0.81514084,
    "HotelName": "Old Century Hotel",
    "Description": "The hotel is situated in a nineteenth
century plaza, which has been expanded and renovated to the highest
architectural standards to create a modern, functional and first-class
hotel in which art and unique historical elements coexist with the most
modern comforts.",
    "Tags": [
        "pool",
        "free wifi",
        "concierge"
    ],
}
]
}

```

## 하이브리드 검색

하이브리드 검색은 단일 검색 요청의 키워드 쿼리와 벡터 쿼리로 구성됩니다. 이 예에서는 벡터 쿼리와 전체 텍스트 검색을 동시에 실행합니다.

- **검색 문자열:** historic hotel walk to restaurants and shopping
- **벡터 쿼리 문자열(수학적 표현으로 벡터화):** classic lodging near running trails,  
eateries, retail

1. POST 요청에 붙여넣어 검색 인덱스를 쿼리합니다. 그런 다음, 요청 보내기를 선택합니다.

## HTTP

```
### Run a hybrid query
POST {{baseUrl}}/indexes/hotels-vector-quickstart/docs/search?api-
version=2023-11-01 HTTP/1.1
Content-Type: application/json
Authorization: Bearer {{token}}


{
  "count": true,
  "search": "historic hotel walk to restaurants and shopping",
  "select": "HotelName, Description",
  "top": 7,
  "vectorQueries": [
    {
      "vector": [ VECTOR OMITTED],
      "k": 7,
      "fields": "DescriptionVector",
      "kind": "vector",
      "exhaustive": true
    }
  ]
}
```

하이브리드 쿼리이므로 결과는 RRF(Reciprocal Rank Fusion)에 따라 순위가 매겨집니다. RRF는 여러 검색 결과의 검색 점수를 평가하고 그 반대를 취한 다음 결합된 결과를 병합하고 정렬합니다. `top` 개의 결과가 반환됩니다.

## 2. 응답을 검토합니다.

### JSON

```
{
  "@odata.count": 7,
  "value": [
    {
      "@search.score": 0.03279569745063782,
      "HotelName": "Luxury Lion Resort",
      "Description": "Unmatched Luxury. Visit our downtown hotel
to indulge in luxury accommodations. Moments from the stadium, we
feature the best in comfort"
    },
    {
      "@search.score": 0.03226646035909653,
      "HotelName": "Sublime Palace Hotel",
      "Description": "Sublime Palace Hotel is located in the
heart of the historic center of Sublime in an extremely vibrant and
lively area within short walking distance to the sites and landmarks of
the city and is surrounded by the extraordinary beauty of churches,
buildings, shops and monuments. Sublime Palace is part of a lovingly
restored 1800 palace."
    },
    ...
  ]
}
```

```

    },
    "@search.score": 0.03226646035909653,
    "HotelName": "Swirling Currents Hotel",
    "Description": "Spacious rooms, glamorous suites and
residences, rooftop pool, walking access to shopping, dining,
entertainment and the city center."
},
{
    "@search.score": 0.03205128386616707,
    "HotelName": "Nordick's Valley Motel",
    "Description": "Only 90 miles (about 2 hours) from the
nation's capital and nearby most everything the historic valley has to
offer. Hiking? Wine Tasting? Exploring the caverns? It's all nearby
and we have specially priced packages to help make our B&B your home
base for fun while visiting the valley."
},
{
    "@search.score": 0.03128054738044739,
    "HotelName": "Gastronomic Landscape Hotel",
    "Description": "The Hotel stands out for its gastronomic
excellence under the management of William Dough, who advises on and
oversees all of the Hotel's restaurant services."
},
{
    "@search.score": 0.03100961446762085,
    "HotelName": "Old Century Hotel",
    "Description": "The hotel is situated in a nineteenth
century plaza, which has been expanded and renovated to the highest
architectural standards to create a modern, functional and first-class
hotel in which art and unique historical elements coexist with the most
modern comforts."
},
{
    "@search.score": 0.03077651560306549,
    "HotelName": "Stay-Kay City Hotel",
    "Description": "The hotel is ideally located on the main
commercial artery of the city in the heart of New York."
}
]
}

```

RRF는 결과를 병합하므로 입력을 검토하는 데 도움이 됩니다. 다음은 전체 텍스트 쿼리에서만 나온 결과입니다. 상위 2개 결과는 숭고한 팰리스 호텔과 히스토리 라이온 리조트입니다. 숭고한 팰리스 호텔은 BM25 관련성 점수가 더 강합니다.

JSON

```

{
    "@search.score": 2.2626662,
    "HotelName": "Sublime Palace Hotel",
    "Description": "Sublime Palace Hotel is located in the
heart of the historic center of Sublime in an extremely vibrant and
lively area within short walking distance to the sites and landmarks of

```

```

the city and is surrounded by the extraordinary beauty of churches,
buildings, shops and monuments. Sublime Palace is part of a lovingly
restored 1800 palace."
},
{
    "@search.score": 0.86421645,
    "HotelName": "Luxury Lion Resort",
    "Description": "Unmatched Luxury. Visit our downtown hotel
to indulge in luxury accommodations. Moments from the stadium, we
feature the best in comfort"
},

```

HNSW를 사용하여 일치 항목을 찾는 벡터 전용 쿼리에서 Sublime Palace Hotel은 네 번째 위치로 떨어집니다. 전체 텍스트 검색에서 2위, 벡터 검색에서 3위를 차지했던 Historic Lion은 동일한 변동 범위를 가지지 않으므로 균질화된 결과 집합에서 상위 일치 항목으로 표시됩니다.

#### JSON

```

"value": [
{
    "@search.score": 0.857736,
    "HotelId": "48",
    "HotelName": "Nordick's Valley Motel",
    "Description": "Only 90 miles (about 2 hours) from the
nation's capital and nearby most everything the historic valley has to
offer. Hiking? Wine Tasting? Exploring the caverns? It's all nearby
and we have specially priced packages to help make our B&B your home
base for fun while visiting the valley.",
    "Category": "Boutique"
},
{
    "@search.score": 0.8399129,
    "HotelId": "49",
    "HotelName": "Swirling Currents Hotel",
    "Description": "Spacious rooms, glamorous suites and
residences, rooftop pool, walking access to shopping, dining,
entertainment and the city center.",
    "Category": "Luxury"
},
{
    "@search.score": 0.8383954,
    "HotelId": "13",
    "HotelName": "Luxury Lion Resort",
    "Description": "Unmatched Luxury. Visit our downtown hotel
to indulge in luxury accommodations. Moments from the stadium, we
feature the best in comfort",
    "Category": "Resort and Spa"
},
{
    "@search.score": 0.8254346,
    "HotelId": "4",
    "HotelName": "Historic Lion Hotel"
}
]
```

```

        "HotelName": "Sublime Palace Hotel",
        "Description": "Sublime Palace Hotel is located in the
heart of the historic center of Sublime in an extremely vibrant and
lively area within short walking distance to the sites and landmarks of
the city and is surrounded by the extraordinary beauty of churches,
buildings, shops and monuments. Sublime Palace is part of a lovingly
restored 1800 palace.",
        "Category": "Boutique"
    },
    {
        "@search.score": 0.82380056,
        "HotelId": "1",
        "HotelName": "Stay-Kay City Hotel",
        "Description": "The hotel is ideally located on the main
commercial artery of the city in the heart of New York.",
        "Category": "Boutique"
    },
    {
        "@search.score": 0.81514084,
        "HotelId": "2",
        "HotelName": "Old Century Hotel",
        "Description": "The hotel is situated in a nineteenth
century plaza, which has been expanded and renovated to the highest
architectural standards to create a modern, functional and first-class
hotel in which art and unique historical elements coexist with the most
modern comforts.",
        "Category": "Boutique"
    },
    {
        "@search.score": 0.8133763,
        "HotelId": "3",
        "HotelName": "Gastronomic Landscape Hotel",
        "Description": "The Hotel stands out for its gastronomic
excellence under the management of William Dough, who advises on and
oversees all of the Hotel's restaurant services.",
        "Category": "Resort and Spa"
    }
]

```

## 필터를 사용한 의미 체계 하이브리드 검색

컬렉션의 마지막 쿼리는 다음과 같습니다. 의미 체계 순위를 사용하는 이 하이브리드 쿼리는 워싱턴 D.C.에서 반경 500km 내에 있는 호텔만 표시하도록 필터링되었습니다. `vectorFilterMode` 를 null로 설정할 수 있습니다. 이는 기본값(최신 인덱스의 경우 `preFilter`, 이전 인덱스의 경우 `postFilter`)과 동일합니다.

1. POST 요청에 붙여넣어 검색 인덱스를 쿼리합니다. 그런 다음, **요청 보내기**를 선택합니다.

HTTP

```

### Run a hybrid query
POST {{baseUrl}}/indexes/hotels-vector-quickstart/docs/search?api-
version=2023-11-01 HTTP/1.1
Content-Type: application/json
Authorization: Bearer {{token}}


{
  "count": true,
  "search": "historic hotel walk to restaurants and shopping",
  "select": "HotelId, HotelName, Category, Description,Address/City,
Address/StateProvince",
  "filter": "geo.distance(Location, geography'POINT(-77.03241
38.90166)') le 500",
  "vectorFilterMode": null,
  "facets": [ "Address/StateProvince"],
  "top": 7,
  "queryType": "semantic",
  "answers": "extractive|count-3",
  "captions": "extractive|highlight-true",
  "semanticConfiguration": "my-semantic-config",
  "vectorQueries": [
    {
      "vector": [ VECTOR OMITTED ],
      "k": 7,
      "fields": "DescriptionVector",
      "kind": "vector",
      "exhaustive": true
    }
  ]
}

```

2. 응답을 검토합니다. 응답은 호텔 세 곳으로, 위치별로 필터링되고 `StateProvince`로 패싯 처리되며 의미 체계 순위가 다시 지정되어 검색 문자열 쿼리(`historic hotel walk to restaurants and shopping`)에 가장 가까운 결과를 승격시킵니다.

소용돌이치는 전류 호텔은 이제 최고의 자리로 이동합니다. 의미 체계 순위없이, 노르딕의 빨리 모텔은 1 위입니다. 의미 체계 순위 지정을 통해 기계 독해 모델은 `historic`이 "식사(식당) 및 쇼핑 위치가 도보 거리 내에 있는 호텔"에 적용된다는 것을 인식합니다.

#### JSON

```
{
  "@odata.count": 3,
  "@search.facets": {
    "Address/StateProvince": [
      {
        "count": 1,
        "value": "NY"
      },

```

```
        {
            "count": 1,
            "value": "VA"
        }
    ],
},
"@search.answers": [],
"value": [
{
    "@search.score": 0.03306011110544205,
    "@search.rerankerScore": 2.5094974040985107,
    "HotelId": "49",
    "HotelName": "Swirling Currents Hotel",
    "Description": "Spacious rooms, glamorous suites and residences, rooftop pool, walking access to shopping, dining, entertainment and the city center.",
    "Category": "Luxury",
    "Address": {
        "City": "Arlington",
        "StateProvince": "VA"
    }
},
{
    "@search.score": 0.03306011110544205,
    "@search.rerankerScore": 2.0370211601257324,
    "HotelId": "48",
    "HotelName": "Nordick's Valley Motel",
    "Description": "Only 90 miles (about 2 hours) from the nation's capital and nearby most everything the historic valley has to offer. Hiking? Wine Tasting? Exploring the caverns? It's all nearby and we have specially priced packages to help make our B&B your home base for fun while visiting the valley.",
    "Category": "Boutique",
    "Address": {
        "City": "Washington D.C.",
        "StateProvince": null
    }
},
{
    "@search.score": 0.032258063554763794,
    "@search.rerankerScore": 1.6706111431121826,
    "HotelId": "1",
    "HotelName": "Stay-Kay City Hotel",
    "Description": "The hotel is ideally located on the main commercial artery of the city in the heart of New York.",
    "Category": "Boutique",
    "Address": {
        "City": "New York",
        "StateProvince": "NY"
    }
}
]
```

## 주요 정보:

- 벡터 검색은 `vectors.value` 속성을 통해 지정됩니다. 키워드 검색은 `search` 속성을 통해 지정됩니다.
- 하이브리드 검색에서는 키워드에 대한 전체 텍스트 검색과 벡터 검색을 통합할 수 있습니다. 필터, 맞춤법 검사 및 의미 체계 순위 지정은 텍스트 콘텐츠에만 적용되며 벡터에는 적용되지 않습니다. 이 마지막 쿼리에는 시스템이 충분히 강력한 쿼리를 생성하지 않았기 때문에 의미 체계 `answer` 가 없습니다.
- 실제 결과에는 의미 체계 캡션 및 강조 표시를 포함한 더 자세한 내용이 포함됩니다. 가독성을 위해 결과가 수정되었습니다. 응답의 전체 구조를 확인하려면 REST 클라이언트에서 요청을 실행하세요.

## 정리

본인 소유의 구독으로 이 모듈을 진행하고 있는 경우에는 프로젝트가 끝날 때 여기에서 만든 리소스가 계속 필요한지 확인하는 것이 좋습니다. 계속 실행되는 리소스에는 요금이 부과될 수 있습니다. 리소스를 개별적으로 삭제하거나 리소스 그룹을 삭제하여 전체 리소스 세트를 삭제할 수 있습니다.

가장 왼쪽 창의 **모든 리소스** 또는 **리소스 그룹** 링크를 사용하여 포털에서 리소스를 찾고 관리할 수 있습니다.

이 `DELETE` 명령을 사용해 볼 수도 있습니다.

```
HTTP

### Delete an index
DELETE {{baseUrl}}/indexes/hotels-vector-quickstart?api-version=2023-11-01
HTTP/1.1
Content-Type: application/json
Authorization: Bearer {{token}}
```

## 다음 단계

다음 단계로 [Python](#), [C#](#) 또는 [JavaScript](#)의 데모 코드를 살펴보는 것이 좋습니다.

## 피드백

이 페이지가 도움이 되었나요?

 Yes

 No



# 빠른 시작: Azure AI 검색의 그라운딩 데이터가 있는 생성형 검색(RAG)

아티클 • 2024. 10. 31.

이 빠른 시작에서는 Azure AI Search에서 인덱싱된 콘텐츠를 통해 대화형 검색 환경을 위해 LLM(대규모 언어 모델)에 기본 및 복잡한 쿼리를 보내는 방법을 보여 줍니다. Azure Portal을 사용하여 리소스를 설정한 다음 Python 코드를 실행하여 API를 호출합니다.

## 필수 구성 요소

- Azure 구독 [체험 계정 만들기](#)
- [의미 순위 매기기](#)를 사용하도록 설정할 수 있도록 Azure AI 검색 기본 계층 이상. 지역은 Azure OpenAI에 사용되는 지역과 동일해야 합니다.
- Azure AI Search와 동일한 지역에 배포 또는 동등한 LLM을 배포 `gpt-4ogpt-4o-mini` 하는 Azure OpenAI [리소스](#)입니다.
- Python 확장  [및 Jupyter 패키지](#)가 있는 [Visual Studio Code](#). 자세한 내용은 [Visual Studio Code의 Python](#)을 참조하세요.

## 파일 다운로드

GitHub에서 [Jupyter Notebook](#)을 다운로드 [하여](#) 이 빠른 시작에서 요청을 보냅니다. 자세한 내용은 [GitHub에서 파일 다운로드](#)를 참조하세요.

이 문서의 지침을 사용하여 로컬 시스템에서 새 파일을 시작하고 수동으로 요청을 만들 수도 있습니다.

## 액세스 구성

검색 엔드포인트에 대한 요청은 인증 및 권한 부여를 받아야 합니다. 이 작업에 API 키 또는 역할을 사용할 수 있습니다. 키는 시작하기가 더 쉽지만 역할이 더 안전합니다. 이 빠른 시작에서는 역할을 가정합니다.

두 개의 클라이언트를 설정하므로 두 리소스 모두에 대한 권한이 필요합니다.

Azure AI 검색은 로컬 시스템에서 쿼리 요청을 수신합니다. 해당 작업에 대한 **검색 인덱스 데이터 읽기 권한자** 역할 할당을 자신에게 할당합니다. 호텔 샘플 인덱스도 만들고 로드하는 경우 **Search Service 기여자** 및 **검색 인덱스 데이터 기여자** 역할도 추가합니다.

Azure OpenAI는 로컬 시스템에서 "몇 가지 호텔을 추천할 수 있나요?"(쿼리)를 수신하고 검색 서비스에서 검색 결과(원본)를 수신합니다. 자신과 검색 서비스에 **Cognitive Services OpenAI 사용자** 역할을 할당합니다.

1. [Azure Portal](#)에 로그인합니다.
2. 역할 할당을 제공하려면 시스템이 할당한 관리 ID를 사용하도록 Azure AI 검색을 구성합니다.
  - a. Azure Portal에서 [검색 서비스를 찾습니다](#).
  - b. 왼쪽 메뉴에서 **설정>ID**를 선택합니다.
  - c. 시스템 할당 탭에서 상태를 **켜짐**으로 설정합니다.
3. 역할 기반 액세스에 대한 Azure AI 검색 구성:
  - a. Azure Portal에서 Azure AI 검색 서비스를 찾습니다.
  - b. 왼쪽 메뉴에서 **설정>키**를 선택한 다음 **역할 기반 액세스 제어** 또는 **둘 다**를 선택합니다.
4. 역할 할당:
  - a. 왼쪽 메뉴에서 **IAM(액세스 제어)**를 선택합니다.
  - b. Azure AI 검색에서 검색 인덱스를 만들고, 로드하고, 쿼리할 수 있는 권한이 있는지 확인합니다.
    - **검색 인덱스 데이터 읽기 권한자**
    - **검색 인덱스 데이터 기여자**
    - **Search 서비스 기여자**
  - c. Azure OpenAI에서 **액세스 제어(IAM)**를 선택하여 자신과 검색 서비스에 Azure OpenAI에 대한 ID 권한을 할당합니다. 이 빠른 시작의 코드는 로컬로 실행됩니다. Azure OpenAI에 대한 요청은 시스템에서 시작됩니다. 또한 검색 엔진의 검색 결과가 Azure OpenAI에 전달됩니다. 이러한 이유로 자신과 검색 서비스 모두에 Azure OpenAI에 대한 권한이 필요합니다.
    - **Cognitive Services OpenAI 사용자**

사용 권한이 적용되는 데 몇 분 정도 걸릴 수 있습니다.

## 인덱스 만들기

몇 분 안에 만들 수 있고 모든 검색 서비스 계층에서 실행되는 hotels-sample-index를 권장합니다. 이 인덱스는 기본 제공 샘플 데이터를 사용하여 생성됩니다.

1. Azure Portal에서 [검색 서비스를 찾습니다](#).
2. **개요** 홈페이지에서 **데이터 가져오기**를 선택하여 마법사를 시작합니다.
3. **데이터에 연결** 페이지의 드롭다운 목록에서 **샘플**을 선택합니다.
4. **hotels-sample**을 선택합니다.
5. 나머지 페이지에서 **다음**을 선택하고 기본값을 적용합니다.
6. 인덱스가 만들어지면 왼쪽 메뉴에서 **Search 관리 > 인덱스**를 선택하여 인덱스를 엽니다.
7. **JSON 편집**을 선택합니다.
8. "의미론적"을 검색하여 의미론적 구성에 대한 인덱스의 섹션을 찾습니다. 비어 있는 `"semantic": {}` 줄을 다음과 같은 의미론적 구성으로 바꿉니다. 이 예제에서는 이 빠른 시작을 실행하는 데 중요한 `"defaultConfiguration"` 을 지정합니다.

JSON

```
"semantic":{  
    "defaultConfiguration":"semantic-config",  
    "configurations": [  
        {  
            "name": "semantic-config",  
            "prioritizedFields": {  
                "titleField": {  
                    "fieldName": "HotelName"  
                },  
                "prioritizedContentFields": [  
                    {  
                        "fieldName": "Description"  
                    }  
                ],  
                "prioritizedKeywordsFields": [  
                    {  
                        "fieldName": "Category"  
                    },  
                    {  
                        "fieldName": "Tags"  
                    }  
                ]  
            }  
        }  
    ]  
},
```

9. 변경 내용을 저장합니다.

10. 검색 탐색기에서 다음 쿼리를 실행하여 인덱스를 테스트합니다. `complimentary breakfast`.

출력은 다음 예제와 비슷해야 합니다. 검색 엔진에서 직접 반환되는 결과는 의미 순 위매기기를 사용하는 경우 의미 체계 순위 지정 점수 및 캡션과 검색 점수와 같은 메타데이터와 함께 필드 및 해당 문자값으로 구성됩니다. Select 문을 사용하여 HotelName, Description 및 Tags 필드만 반환했습니다.

```
{
  "@odata.count": 18,
  "@search.answers": [],
  "value": [
    {
      "@search.score": 2.2896252,
      "@search.rerankerScore": 2.506816864013672,
      "@search.captions": [
        {
          "text": "Head Wind Resort. Suite. coffee in lobby\r\nfree wifi\r\nview. The best of old town hospitality combined with views of the river and cool breezes off the prairie. Our penthouse suites offer views for miles and the rooftop plaza is open to all guests from sunset to 10 p.m. Enjoy a **complimentary continental breakfast** in the lobby, and free Wi-Fi throughout the hotel..",
          "highlights": ""
        }
      ],
      "HotelName": "Head Wind Resort",
      "Description": "The best of old town hospitality combined with views of the river and cool breezes off the prairie. Our penthouse suites offer views for miles and the rooftop plaza is open to all guests from sunset to 10 p.m. Enjoy a complimentary continental breakfast in the lobby, and free Wi-Fi throughout the hotel.",
      "Tags": [
        "coffee in lobby",
        "free wifi",
        "view"
      ]
    },
    {
      "@search.score": 2.2158256,
      "@search.rerankerScore": 2.288334846496582,
      "@search.captions": [
        {
          "text": "Swan Bird Lake Inn. Budget. continental breakfast\r\nfree wifi\r\n24-hour front desk service. We serve a continental-style breakfast each morning, featuring a variety of food and drinks. Our locally made, oh-so-soft, caramel cinnamon rolls are a favorite with our guests. Other breakfast items include coffee, orange juice, and a selection of fruits and pastries."
        }
      ]
    }
  ]
}
```

```

        juice, milk, cereal, instant oatmeal, bagels, and muffins..",
        "highlights": ""
    }
],
{
    "HotelName": "Swan Bird Lake Inn",
    "Description": "We serve a continental-style breakfast each morning, featuring a variety of food and drinks. Our locally made, oh-so-soft, caramel cinnamon rolls are a favorite with our guests. Other breakfast items include coffee, orange juice, milk, cereal, instant oatmeal, bagels, and muffins.",
    "Tags": [
        "continental breakfast",
        "free wifi",
        "24-hour front desk service"
    ]
},
{
    "@search.score": 0.92481667,
    "@search.rerankerScore": 2.221315860748291,
    "@search.captions": [
    {
        "text": "White Mountain Lodge & Suites. Resort and Spa. continental breakfast\r\nnpool\r\nrestaurant. Live amongst the trees in the heart of the forest. Hike along our extensive trail system. Visit the Natural Hot Springs, or enjoy our signature hot stone massage in the Cathedral of Firs. Relax in the meditation gardens, or join new friends around the communal firepit. Weekend evening entertainment on the patio features special guest musicians or poetry readings..",
        "highlights": ""
    }
],
{
    "HotelName": "White Mountain Lodge & Suites",
    "Description": "Live amongst the trees in the heart of the forest. Hike along our extensive trail system. Visit the Natural Hot Springs, or enjoy our signature hot stone massage in the Cathedral of Firs. Relax in the meditation gardens, or join new friends around the communal firepit. Weekend evening entertainment on the patio features special guest musicians or poetry readings.",
    "Tags": [
        "continental breakfast",
        "pool",
        "restaurant"
    ]
},
...
]
}

```

## 서비스 엔드포인트 가져오기

나머지 섹션에서는 Azure OpenAI 및 Azure AI 검색에 대한 API 호출을 설정합니다. 서비스 엔드포인트를 가져와서 코드에서 변수로 제공할 수 있습니다.

1. Azure Portal [에](#) 로그인합니다.
2. 검색 서비스 찾기 [.](#)
3. **개요** 홈페이지에서 URL을 복사합니다. 엔드포인트의 예는 다음과 같습니다.  
`https://example.search.windows.net`
4. Azure OpenAI 서비스를 찾습니다 [.](#)

5. **개요** 홈페이지에서 엔드포인트를 볼 링크를 선택합니다. URL을 복사합니다. 엔드포인트의 예는 다음과 같습니다. `https://example.openai.azure.com/`

## 쿼리 및 채팅 스레드 설정

이 섹션에서는 Visual Studio Code와 Python을 사용하여 Azure OpenAI에서 채팅 완료 API를 호출합니다.

1. Visual Studio Code를 시작하고 [.ipynb 파일을 열거나](#) 새 Python 파일을 만듭니다.
2. 다음 Python 패키지를 설치합니다.

```
Python

! pip install azure-search-documents==11.6.0b5 --quiet
! pip install azure-identity==1.16.1 --quiet
! pip install openai --quiet
! pip install aiohttp --quiet
! pip install ipykernel --quiet
```

3. 다음 변수를 설정하여 자리 표시자를 이전 단계에서 수집한 엔드포인트로 대체합니다.

```
Python

AZURE_SEARCH_SERVICE: str = "PUT YOUR SEARCH SERVICE ENDPOINT HERE"
AZURE_OPENAI_ACCOUNT: str = "PUT YOUR AZURE OPENAI ENDPOINT HERE"
AZURE_DEPLOYMENT_MODEL: str = "gpt-4o"
```

4. 클라이언트, 프롬프트, 쿼리 및 응답을 설정합니다.

```
Python

# Set up the query for generating responses
from azure.identity import DefaultAzureCredential
from azure.identity import get_bearer_token_provider
from azure.search.documents import SearchClient
```

```
from openai import AzureOpenAI

credential = DefaultAzureCredential()
token_provider = get_bearer_token_provider(credential,
"https://cognitiveservices.azure.com/.default")
openai_client = AzureOpenAI(
    api_version="2024-06-01",
    azure_endpoint=AZURE_OPENAI_ACCOUNT,
    azure_ad_token_provider=token_provider
)

search_client = SearchClient(
    endpoint=AZURE_SEARCH_SERVICE,
    index_name="hotels-sample-index",
    credential=credential
)

# This prompt provides instructions to the model
GROUNDED_PROMPT="""

You are a friendly assistant that recommends hotels based on activities and amenities.



Answer the query using only the sources provided below in a friendly and concise bulleted manner.



Answer ONLY with the facts listed in the list of sources below.



If there isn't enough information below, say you don't know.



Do not generate answers that don't use the sources below.



Query: {query}



Sources:\n{sources}

"""

# Query is the question being asked. It's sent to the search engine and the LLM.
query="Can you recommend a few hotels with complimentary breakfast?"

# Set up the search results and the chat thread.
# Retrieve the selected fields from the search index related to the question.
search_results = search_client.search(
    search_text=query,
    top=5,
    select="Description,HotelName,Tags"
)
sources_formatted = "\n".join([f'{document["HotelName"]}: {document["Description"]}:{document["Tags"]}' for document in search_results])

response = openai_client.chat.completions.create(
    messages=[
        {
            "role": "user",
            "content": GROUNDED_PROMPT.format(query=query,
sources=sources_formatted)
        }
    ],
    model=AZURE_DEPLOYMENT_MODEL
```

```
)  
print(response.choices[0].message.content)
```

출력은 Azure OpenAI의 결과이며 여러 호텔에 대한 권장 사항으로 구성됩니다. 다음은 출력의 모양에 대한 예입니다.

```
Sure! Here are a few hotels that offer complimentary breakfast:  
  
- **Head Wind Resort**  
- Complimentary continental breakfast in the lobby  
- Free Wi-Fi throughout the hotel  
  
- **Double Sanctuary Resort**  
- Continental breakfast included  
  
- **White Mountain Lodge & Suites**  
- Continental breakfast available  
  
- **Swan Bird Lake Inn**  
- Continental-style breakfast each morning with a variety of food and drinks  
such as caramel cinnamon rolls, coffee, orange juice, milk, cereal,  
instant oatmeal, bagels, and muffins
```

**사용할 수 없음** 오류 메시지가 표시되면 Azure AI 검색 구성을 확인하여 역할 기반 액세스가 사용하도록 설정되어 있는지 확인합니다.

**권한 부여 실패** 오류 메시지가 표시되면 몇 분 정도 기다렸다가 다시 시도하세요. 역할 할당이 작동하려면 몇 분 정도 걸릴 수 있습니다.

그렇지 않은 경우 더 실험하려면 쿼리를 변경하고 마지막 단계를 다시 실행하여 모델이 기초 데이터를 사용하여 작동하는 방식을 더 잘 이해합니다.

프롬프트를 수정하여 출력의 톤이나 구조를 변경할 수도 있습니다.

쿼리 매개 변수 단계에서 `use_semantic_reranker=False`를 설정하여 의미론적 순위 지정 없이 쿼리를 시도할 수도 있습니다. 의미론적 순위 지정은 쿼리 결과의 관련성과 LLM이 유용한 정보를 반환하는 기능을 눈에 띄게 개선할 수 있습니다. 실험을 통해 콘텐츠에 변화를 가져올지 여부를 결정할 수 있습니다.

## 복잡한 RAG 쿼리 보내기

Azure AI Search는 중첩된 JSON 구조에 대한 복합 형식을 지원합니다. hotels-sample-index에서, Address, Address.StateProvince, Address.PostalCode 및 Address.Country로 구성된 복합 형식의 Address.StreetAddress, Address.City 예입니다. 인덱스는 각 호텔에 대한 복잡한 컬렉션 Rooms도 가지고 있습니다.

인덱스 형식이 복잡한 경우 검색 결과 출력을 JSON으로 변환한 다음 JSON을 LLM에 전달하는 경우 쿼리에서 해당 필드를 제공할 수 있습니다. 다음 예제에서는 요청에 복합 형식을 추가합니다. 서식 지정 지침에는 JSON 사양이 포함됩니다.

Python

```
import json

# Query is the question being asked. It's sent to the search engine and the
# LLM.
query="Can you recommend a few hotels that offer complimentary breakfast?
Tell me their description, address, tags, and the rate for one room that
sleeps 4 people."

# Set up the search results and the chat thread.
# Retrieve the selected fields from the search index related to the
# question.
selected_fields = ["HotelName", "Description", "Address", "Rooms", "Tags"]
search_results = search_client.search(
    search_text=query,
    top=5,
    select=selected_fields,
    query_type="semantic"
)
sources_filtered = [{field: result[field] for field in selected_fields} for
result in search_results]
sources_formatted = "\n".join([json.dumps(source) for source in
sources_filtered])

response = openai_client.chat.completions.create(
    messages=[
        {
            "role": "user",
            "content": GROUNDED_PROMPT.format(query=query,
sources=sources_formatted)
        }
    ],
    model=AZURE_DEPLOYMENT_MODEL
)

print(response.choices[0].message.content)
```

출력은 Azure OpenAI의 출력이며 복잡한 형식의 컨텐츠를 추가합니다.

Here are a few hotels that offer complimentary breakfast and have rooms that sleep 4 people:

1. \*\*Head Wind Resort\*\*

- \*\*Description:\*\* The best of old town hospitality combined with views of the river and cool breezes off the prairie. Enjoy a complimentary continental breakfast in the lobby, and free Wi-Fi throughout the hotel.
- \*\*Address:\*\* 7633 E 63rd Pl, Tulsa, OK 74133, USA
- \*\*Tags:\*\* Coffee in lobby, free Wi-Fi, view
- \*\*Room for 4:\*\* Suite, 2 Queen Beds (Amenities) - \$254.99

2. \*\*Double Sanctuary Resort\*\*

- \*\*Description:\*\* 5-star Luxury Hotel - Biggest Rooms in the city. #1 Hotel in the area listed by Traveler magazine. Free WiFi, Flexible check in/out, Fitness Center & espresso in room. Offers continental breakfast.
- \*\*Address:\*\* 2211 Elliott Ave, Seattle, WA 98121, USA
- \*\*Tags:\*\* View, pool, restaurant, bar, continental breakfast
- \*\*Room for 4:\*\* Suite, 2 Queen Beds (Amenities) - \$254.99

3. \*\*Swan Bird Lake Inn\*\*

- \*\*Description:\*\* Continental-style breakfast featuring a variety of food and drinks. Locally made caramel cinnamon rolls are a favorite.
- \*\*Address:\*\* 1 Memorial Dr, Cambridge, MA 02142, USA
- \*\*Tags:\*\* Continental breakfast, free Wi-Fi, 24-hour front desk service
- \*\*Room for 4:\*\* Budget Room, 2 Queen Beds (City View) - \$85.99

4. \*\*Gastronomic Landscape Hotel\*\*

- \*\*Description:\*\* Known for its culinary excellence under the management of William Dough, offers continental breakfast.
  - \*\*Address:\*\* 3393 Peachtree Rd, Atlanta, GA 30326, USA
  - \*\*Tags:\*\* Restaurant, bar, continental breakfast
  - \*\*Room for 4:\*\* Budget Room, 2 Queen Beds (Amenities) - \$66.99
- ...
- \*\*Tags:\*\* Pool, continental breakfast, free parking
  - \*\*Room for 4:\*\* Budget Room, 2 Queen Beds (Amenities) - \$60.99

Enjoy your stay! Let me know if you need any more information.

## 오류 문제 해결

인증 오류를 디버그하려면 검색 엔진 및 LLM을 호출하는 단계 앞에 다음 코드를 삽입합니다.

```
import sys
import logging # Set the logging level for all azure-storage-* libraries
logger = logging.getLogger('azure.identity')
logger.setLevel(logging.DEBUG)

handler = logging.StreamHandler(stream=sys.stdout)
formatter = logging.Formatter('[%(levelname)s %(name)s] %(message)s')
handler.setFormatter(formatter)
logger.addHandler(handler)
```

쿼리 스크립트를 다시 실행합니다. 이제 문제에 대한 자세한 정보를 제공하는 INFO 및 DEBUG 문을 출력에 가져와야 합니다.

ManagedIdentityCredential 및 토큰 획득 실패와 관련된 출력 메시지가 표시되면 테넌트가 여러 개 있고 Azure 로그인에서 검색 서비스가 없는 테넌트를 사용하고 있을 수 있습니다. 테넌트 ID를 가져오려면 Azure Portal에서 "테넌트 속성"을 검색하거나 `az login tenant list`을(를) 실행합니다.

테넌트 ID가 있으면 명령 프롬프트에서 `az login --tenant <YOUR-TENANT-ID>`을(를) 실행한 다음 스크립트를 다시 실행합니다.

## 정리

본인 소유의 구독으로 이 모듈을 진행하고 있는 경우에는 프로젝트가 끝날 때 여기에서 만든 리소스가 계속 필요한지 확인하는 것이 좋습니다. 계속 실행되는 리소스에는 요금이 부과될 수 있습니다. 리소스를 개별적으로 삭제하거나 리소스 그룹을 삭제하여 전체 리소스 세트를 삭제할 수 있습니다.

가장 왼쪽 창의 **모든 리소스** 또는 **리소스 그룹** 링크를 사용하여 포털에서 리소스를 찾고 관리할 수 있습니다.

## 참고 항목

- [튜토리얼: Azure AI에서 RAG 솔루션을 빌드하는 방법 검색](#)

## 피드백

이 페이지가 도움이 되었나요?

 Yes

 No

# 빠른 시작: Azure SDK를 사용하여 전체 텍스트 검색

아티클 • 2024. 11. 01.

Azure SDK의 `Azure.Search.Documents` 클라이언트 라이브러리를 사용하여 [전체 텍스트 검색](#)용 샘플 데이터를 사용하여 검색 인덱스 만들고 로드하며 쿼리하는 방법을 알아봅니다. 전체 텍스트 검색은 인덱싱 및 쿼리에 Apache Lucene을 사용하고 BM25 순위 지정 알고리즘을 사용하여 결과를 채점합니다.

이 빠른 시작에는 다음 SDK에 대한 단계가 있습니다.

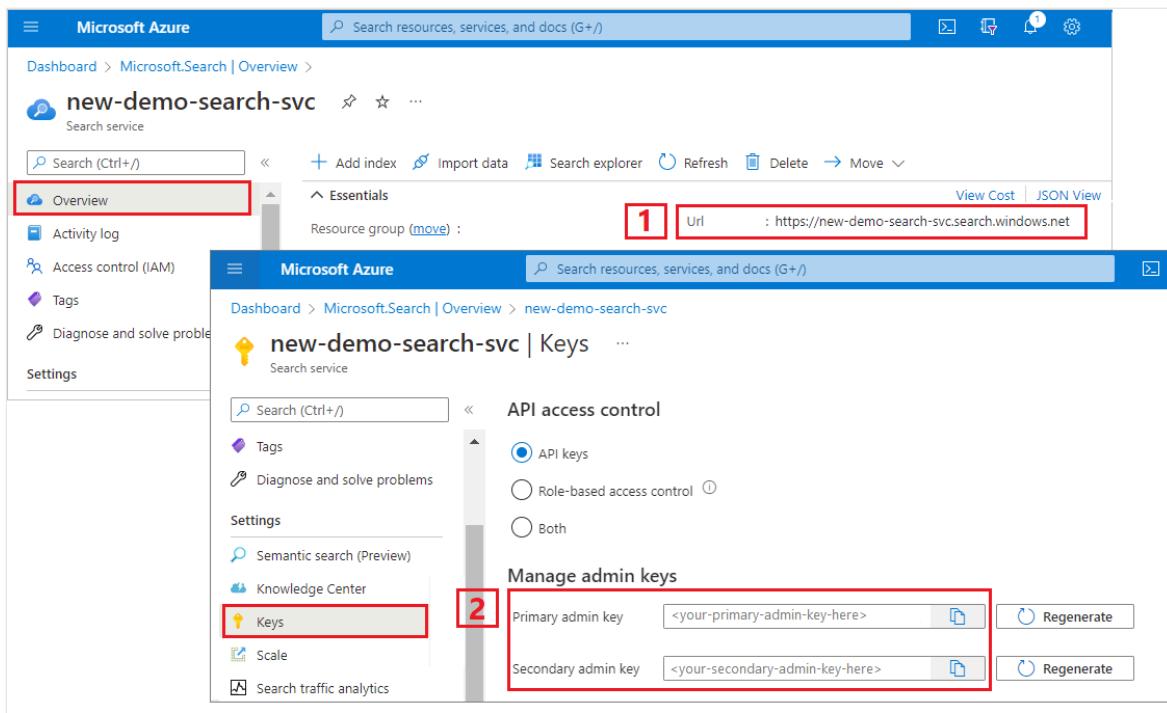
- [.NET용 Azure SDK](#)
- [Python용 Azure SDK](#)
- [Java용 Azure SDK](#)
- [JavaScript/TypeScript용 Azure SDK](#)

## 필수 조건

- 활성 구독이 있는 Azure 계정. 무료로 [계정을 만들 수 있습니다](#).
- Azure AI 검색 서비스. 서비스가 없으면 [서비스를 만듭니다](#). 이 빠른 시작에서는 무료 계층을 사용할 수 있습니다.
- 서비스에 대한 API 키 및 서비스 엔드포인트입니다. [Azure Portal](#)에 로그인하고 [검색 서비스를 찾습니다](#).

**개요** 섹션에서 URL을 복사하고 이후 단계를 위해 텍스트 편집기로 저장합니다. 엔드포인트의 예는 다음과 같습니다. `https://mydemo.search.windows.net`

**설정>키** 섹션에서 관리자 키를 복사하여 저장하여 개체를 만들고 삭제할 수 있는 모든 권한을 갖습니다. 상호 교환 가능한 기본 키와 보조 키가 있습니다. 둘 중 하나를 선택합니다.



## 인덱스 만들기, 로드 및 쿼리

다음 단계에 사용할 프로그래밍 언어를 선택합니다. `Azure.Search.Documents` 클라이언트 라이브러리는 .NET, Python, Java 및 JavaScript/TypeScript용 Azure SDK에서 사용할 수 있습니다.

### .NET

`Azure.Search.Documents` 클라이언트 라이브러리를 사용하여 콘솔 애플리케이션을 빌드하여 검색 인덱스를 만들고 로드하며 쿼리합니다.

또는 [소스 코드를 다운로드](#)하여 완성된 프로젝트로 시작하거나 다음 단계를 수행하여 새 프로젝트를 직접 만들 수 있습니다.

## 환경 설정

1. Visual Studio를 시작하고 콘솔 앱용 새 프로젝트를 만듭니다.
2. 도구>NuGet 패키지 관리자에서 솔루션의 NuGet 패키지 관리...를 선택합니다.
3. 찾아보기를 선택합니다.
4. [Azure.Search.Documents 패키지](#)를 검색하고 버전 11.0 이상을 선택합니다.
5. 설치를 선택하여 프로젝트 및 솔루션에 어셈블리를 추가합니다.

## 검색 클라이언트 만들기

1. Program.cs에서 네임스페이스를 AzureSearch.SDK.Quickstart.v11로 변경하고 다음 `using` 지시문을 추가합니다.

```
C#  
  
using Azure;  
using Azure.Search.Documents;  
using Azure.Search.Documents.Indexes;  
using Azure.Search.Documents.Indexes.Models;  
using Azure.Search.Documents.Models;
```

2. 다음 코드를 복사하여 두 개의 클라이언트를 만듭니다. `SearchIndexClient`는 인덱스를 만들고, `SearchClient`는 기존 인덱스를 로드하고 쿼리합니다. 둘 다 만들기/삭제 권한 인증을 위한 서비스 엔드포인트와 관리 API 키가 필요합니다.

코드가 URI를 작성하므로 속성에 `serviceName` 검색 서비스 이름만 지정합니다.

```
C#  
  
static void Main(string[] args)  
{  
    string serviceName = "<your-search-service-name>";  
    string apiKey = "<your-search-service-admin-api-key>";  
    string indexName = "hotels-quickstart";  
  
    // Create a SearchIndexClient to send create/delete index  
    // commands  
    Uri serviceEndpoint = new  
    Uri($"https://{{serviceName}}.search.windows.net/");  
    AzureKeyCredential credential = new  
    AzureKeyCredential(apiKey);  
    SearchIndexClient adminClient = new  
    SearchIndexClient(serviceEndpoint, credential);  
  
    // Create a SearchClient to load and query documents  
    SearchClient srchclient = new SearchClient(serviceEndpoint,  
    indexName, credential);  
    . . .  
}
```

## 인덱스 만들기

이 빠른 시작에서는 호텔 데이터와 함께 로드하고 쿼리를 실행할 호텔 인덱스를 작성합니다. 이 단계에서는 인덱스의 필드를 정의합니다. 각 필드 정의에는 이름, 데이터 형식 및 필드가 사용되는 방식을 결정하는 특성이 포함됩니다.

이 예제에서는 단순성과 가독성을 위해 Azure.Search.Documents 라이브러리의 동기 메서드를 사용합니다. 그러나 프로덕션 시나리오의 경우 비동기 메서드를 사용하여 앱의 확장성 및 응답성을 유지해야 합니다. 예를 들어 [CreateIndex](#) 대신 [CreateIndexAsync](#)를 사용해야 합니다.

1. 다음과 같이 프로젝트에 빈 클래스 정의를 추가합니다. *Hotel.cs*
2. *Hotel.cs*에 다음 코드를 복사하여 호텔 문서의 구조를 정의합니다. 필드의 특성은 필드가 애플리케이션에서 사용되는 방식을 결정합니다. 예를 들어 `IsFilterable` 특성은 필터 식을 지원하는 모든 필드에 할당해야 합니다.

C#

```
using System;
using System.Text.Json.Serialization;
using Azure.Search.Documents.Indexes;
using Azure.Search.Documents.Indexes.Models;

namespace AzureSearch.Quickstart
{
    public partial class Hotel
    {
        [SimpleField(IsKey = true, IsFilterable = true)]
        public string HotelId { get; set; }

        [SearchableField(IsSortable = true)]
        public string HotelName { get; set; }

        [SearchableField(AnalyzerName =
LexicalAnalyzerName.Values.EnLucene)]
        public string Description { get; set; }

        [SearchableField(AnalyzerName =
LexicalAnalyzerName.Values.FrLucene)]
        [JsonPropertyName("Description_fr")]
        public string DescriptionFr { get; set; }

        [SearchableField(IsFilterable = true, IsSortable = true,
IsFacetable = true)]
        public string Category { get; set; }

        [SearchableField(IsFilterable = true, IsFacetable = true)]
        public string[] Tags { get; set; }

        [SimpleField(IsFilterable = true, IsSortable = true,
IsFacetable = true)]
        public bool? ParkingIncluded { get; set; }

        [SimpleField(IsFilterable = true, IsSortable = true,
IsFacetable = true)]
        public DateTimeOffset? LastRenovationDate { get; set; }
    }
}
```

```

        [SimpleField(IsFilterable = true, IsSortable = true,
IsFacetable = true)]
        public double? Rating { get; set; }

        [SearchableField]
        public Address Address { get; set; }
    }
}

```

*Azure.Search.Documents* 클라이언트 라이브러리에서 `SearchableField` 및 `SimpleField`을 사용하여 필드 정의를 간소화할 수 있습니다. 둘 다 `SearchField`의 파생물이며 잠재적으로 코드를 단순화할 수 있습니다.

- `SimpleField`는 모든 데이터 형식일 수 있으며, 항상 검색할 수 없지만(전체 텍스트 검색 쿼리에서 무시됨) 조회할 수는 있습니다(숨겨지지 않음). 다른 특성은 기본적으로 꺼져 있지만 사용하도록 설정할 수 있습니다. 필터, 패싯 또는 점수 매기기 프로필에만 사용되는 문서 ID 또는 필드에는 `SimpleField`를 사용할 수 있습니다. 그렇다면 문서 ID에 대한 `IsKey = true`와 같이 시나리오에 필요한 특성을 적용해야 합니다. 자세한 내용은 소스 코드의 [SimpleFieldAttribute.cs](#)를 참조하세요.
- `SearchableField`는 문자열이어야 하며, 항상 검색할 수 있고 조회할 수 있습니다. 다른 특성은 기본적으로 꺼져 있지만 사용하도록 설정할 수 있습니다. 이 필드 형식은 검색할 수 있으므로 동의어와 분석기 속성의 전체 보충을 지원합니다. 자세한 내용은 소스 코드의 [SearchableFieldAttribute.cs](#)를 참조하세요.

기본 `SearchField` API를 사용하든지 도우미 모델 중 하나를 사용하든지 간에 필터, 패싯 및 정렬 특성을 명시적으로 사용하도록 설정해야 합니다. 예를 들어 이전 [샘플과 같이](#) `IsFilterable`, `IsSortable` 및 `IsFacetable`은 명시적으로 특성을 지정해야 합니다.

3. 두 번째 빈 클래스 정의를 프로젝트에 추가합니다. `Address.cs`. 다음 코드를 클래스에 복사합니다.

```

C#

using Azure.Search.Documents.Indexes;

namespace AzureSearch.Quickstart
{
    public partial class Address
    {
        [SearchableField(IsFilterable = true)]
        public string StreetAddress { get; set; }
    }
}

```

```

        [SearchableField(IsFilterable = true, IsSortable = true,
IsFacetable = true)]
        public string City { get; set; }

        [SearchableField(IsFilterable = true, IsSortable = true,
IsFacetable = true)]
        public string StateProvince { get; set; }

        [SearchableField(IsFilterable = true, IsSortable = true,
IsFacetable = true)]
        public string PostalCode { get; set; }

        [SearchableField(IsFilterable = true, IsSortable = true,
IsFacetable = true)]
        public string Country { get; set; }
    }
}

```

4. 재정의에 대한 `Hotel.Methods.cs` 및 `Address.Methods.cs` 두 개의 클래스를 `ToString()` 더 만듭니다. 이러한 클래스는 콘솔 출력에서 검색 결과를 렌더링 하는 데 사용됩니다. 이러한 클래스의 콘텐츠는 이 문서에서 제공되지 않지만 [GitHub의 파일](#)에서 코드를 복사할 수 있습니다.

5. `Program.cs`에서 `SearchIndex` 개체를 만든 다음, `CreateIndex` 메서드를 호출하여 검색 서비스에서 인덱스를 표시합니다. 인덱스에는 지정된 필드에서 자동 완성을 사용하도록 설정하는 `SearchSuggester`도 포함되어 있습니다.

```

C#

// Create hotels-quickstart index
private static void CreateIndex(string indexName,
SearchIndexClient adminClient)
{
    FieldBuilder fieldBuilder = new FieldBuilder();
    var searchFields = fieldBuilder.Build(typeof(Hotel));

    var definition = new SearchIndex(indexName, searchFields);

    var suggester = new SearchSuggester("sg", new[] { "HotelName",
"Category", "Address/City", "Address/StateProvince" });
    definition.Suggesters.Add(suggester);

    adminClient.CreateOrUpdateIndex(definition);
}

```

## 문서 로드

Azure AI 검색은 서비스에 저장된 콘텐츠를 검색합니다. 이 단계에서는 방금 만든 호텔 인덱스를 따르는 JSON 문서를 로드합니다.

Azure AI 검색에서 검색 문서는 인덱싱에 대한 입력과 쿼리의 출력 모두에 해당하는 데이터 구조입니다. 외부 데이터 소스에서 가져온, 문서 입력은 데이터베이스의 행, Blob Storage의 Blob 또는 디스크의 JSON 문서일 수 있습니다. 이 예에서는 손쉬운 방법을 사용하여 4개 호텔에 대한 JSON 문서를 코드 자체에 포함합니다.

문서를 업로드할 때 [IndexDocumentsBatch](#) 개체를 사용해야 합니다.

[IndexDocumentsBatch](#) 개체에는 [Actions](#) 컬렉션이 포함되며 컬렉션마다 수행할 작업 ([upload](#), [merge](#), [delete](#) 및 [mergeOrUpload](#))을 Azure AI 검색에 알려주는 속성과 문서가 포함됩니다.

1. *Program.cs*에서 문서 및 인덱스 작업의 배열을 만든 다음, 배열을 [IndexDocumentsBatch](#)에 전달합니다. 다음 문서는 호텔 클래스에서 정의한 대로 호텔 빠른 시작 인덱스(호텔 빠른 시작 인덱스)를 준수합니다.

```
C#  
  
// Upload documents in a single Upload request.  
private static void UploadDocuments(SearchClient searchClient)  
{  
    IndexDocumentsBatch<Hotel> batch = IndexDocumentsBatch.Create(  
        IndexDocumentsAction.Upload(  
            new Hotel()  
            {  
                HotelId = "1",  
                HotelName = "Stay-Kay City Hotel",  
                Description = "The hotel is ideally located on the  
main commercial artery of the city in the heart of New York. A few  
minutes away is Time's Square and the historic centre of the city,  
as well as other places of interest that make New York one of  
America's most attractive and cosmopolitan cities.",  
                DescriptionFr = "L'hôtel est idéalement situé sur  
la principale artère commerciale de la ville en plein cœur de New  
York. A quelques minutes se trouve la place du temps et le centre  
historique de la ville, ainsi que d'autres lieux d'intérêt qui font  
de New York l'une des villes les plus attractives et cosmopolites  
de l'Amérique.",  
                Category = "Boutique",  
                Tags = new[] { "pool", "air conditioning",  
"concierge" },  
                ParkingIncluded = false,  
                LastRenovationDate = new DateTimeOffset(1970, 1,  
18, 0, 0, 0, TimeSpan.Zero),  
                Rating = 3.6,  
                Address = new Address()  
                {  
                    StreetAddress = "677 5th Ave",  
                    City = "New York",  
                    State = "NY",  
                    PostalCode = "10019",  
                    Country = "USA"  
                }  
            }  
        )  
    );  
    await batch.UploadAsync(searchClient);  
}
```

```

        StateProvince = "NY",
        PostalCode = "10022",
        Country = "USA"
    }
},
IndexDocumentsAction.Upload(
    new Hotel()
{
    HotelId = "2",
    HotelName = "Old Century Hotel",
    Description = "The hotel is situated in a
nineteenth century plaza, which has been expanded and renovated to
the highest architectural standards to create a modern, functional
and first-class hotel in which art and unique historical elements
coexist with the most modern comforts.",
    DescriptionFr = "L'hôtel est situé dans une place
du XIXe siècle, qui a été agrandie et rénovée aux plus hautes
normes architecturales pour créer un hôtel moderne, fonctionnel et
de première classe dans lequel l'art et les éléments historiques
uniques coexistent avec le confort le plus moderne.",
    Category = "Boutique",
    Tags = new[] { "pool", "free wifi", "concierge" },
    ParkingIncluded = false,
    LastRenovationDate = new DateTimeOffset(1979, 2,
18, 0, 0, 0, TimeSpan.Zero),
    Rating = 3.60,
    Address = new Address()
    {
        StreetAddress = "140 University Town Center
Dr",
        City = "Sarasota",
        StateProvince = "FL",
        PostalCode = "34243",
        Country = "USA"
    }
),
IndexDocumentsAction.Upload(
    new Hotel()
{
    HotelId = "3",
    HotelName = "Gastronomic Landscape Hotel",
    Description = "The Hotel stands out for its
gastronomic excellence under the management of William Dough, who
advises on and oversees all of the Hotel's restaurant services.",
    DescriptionFr = "L'hôtel est situé dans une place
du XIXe siècle, qui a été agrandie et rénovée aux plus hautes
normes architecturales pour créer un hôtel moderne, fonctionnel et
de première classe dans lequel l'art et les éléments historiques
uniques coexistent avec le confort le plus moderne.",
    Category = "Resort and Spa",
    Tags = new[] { "air conditioning", "bar",
"continental breakfast" },
    ParkingIncluded = true,
    LastRenovationDate = new DateTimeOffset(2015, 9,
20, 0, 0, 0, TimeSpan.Zero),

```

```

        Rating = 4.80,
        Address = new Address()
        {
            StreetAddress = "3393 Peachtree Rd",
            City = "Atlanta",
            StateProvince = "GA",
            PostalCode = "30326",
            Country = "USA"
        }
    )),
    IndexDocumentsAction.Upload(
        new Hotel()
    {
        HotelId = "4",
        HotelName = "Sublime Palace Hotel",
        Description = "Sublime Palace Hotel is located in  
the heart of the historic center of Sublime in an extremely vibrant  
and lively area within short walking distance to the sites and  
landmarks of the city and is surrounded by the extraordinary beauty  
of churches, buildings, shops and monuments. Sublime Palace is part  
of a lovingly restored 1800 palace.",
        DescriptionFr = "Le Sublime Palace Hotel est situé  
au coeur du centre historique de sublime dans un quartier  
extrêmement animé et vivant, à courte distance de marche des sites  
et monuments de la ville et est entouré par l'extraordinaire beauté  
des églises, des bâtiments, des commerces et Monuments. Sublime  
Palace fait partie d'un Palace 1800 restauré avec amour.",
        Category = "Boutique",
        Tags = new[] { "concierge", "view", "24-hour front  
desk service" },
        ParkingIncluded = true,
        LastRenovationDate = new DateTimeOffset(1960, 2,  
06, 0, 0, TimeSpan.Zero),
        Rating = 4.60,
        Address = new Address()
        {
            StreetAddress = "7400 San Pedro Ave",
            City = "San Antonio",
            StateProvince = "TX",
            PostalCode = "78216",
            Country = "USA"
        }
    })
);

try
{
    IndexDocumentsResult result =
searchClient.IndexDocuments(batch);
}
catch (Exception)
{
    // If for some reason any documents are dropped during
    // indexing, you can compensate by delaying and
    // retrying. This simple demo just logs the failed document
}

```

```

keys and continues.

Console.WriteLine("Failed to index some of the documents:
{0}");
}
}

```

[IndexDocumentsBatch](#) 개체를 초기화 한 후에는 [SearchClient](#) 개체에서 [IndexDocuments](#)를 호출하여 이 개체를 인덱스에 전송할 수 있습니다.

2. `Main()`에 다음 줄을 추가합니다. 문서 로드는 [SearchClient](#)를 사용하여 수행되지만 작업에는 일반적으로 [SearchIndexClient](#)와 관련된 서비스에 대한 관리자 권한도 있어야 합니다. 이 작업을 설정하는 한 가지 방법은 [SearchClient](#)를 통해 [SearchIndexClient](#) 가져오는 것입니다(`adminClient` 이 예제에서는).

C#

```

SearchClient ingestorClient =
adminClient.GetSearchClient(indexName);

// Load documents
Console.WriteLine("{0}", "Uploading documents...\n");
UploadDocuments(ingesterClient);

```

3. 이 앱은 모든 명령을 순차적으로 실행하는 콘솔 앱이므로 인덱싱과 쿼리 사이에 2초 대기 시간을 추가합니다.

C#

```

// Wait 2 seconds for indexing to complete before starting queries
//(for demo and console-app purposes only)
Console.WriteLine("Waiting for indexing...\n");
System.Threading.Thread.Sleep(2000);

```

2초 지연은 비동기식 인덱싱을 보정합니다. 따라서 쿼리가 실행되기 전에 모든 문서를 인덱싱할 수 있습니다. 지연 시 코딩은 일반적으로 데모, 테스트, 샘플 애플리케이션에서만 필요합니다.

## 인덱스 검색

첫 번째 문서의 인덱싱이 완료되는 즉시 쿼리 결과를 얻을 수 있지만 인덱스에 대한 실제 테스트는 모든 문서의 인덱싱이 완료될 때까지 기다려야 합니다.

이 섹션에서는 쿼리 논리 및 결과라는 두 가지 기능을 추가합니다. 쿼리에는 [Search](#) 메서드를 사용합니다. 이 메서드는 검색 텍스트(쿼리 문자열)뿐 아니라 다른 [옵션](#)을 사용합니다.

SearchResults 클래스는 결과를 나타냅니다.

1. Program.cs 검색 결과를 콘솔에 출력하는 메서드를 만듭니다 WriteDocuments 다.

```
C#  
  
// Write search results to console  
private static void WriteDocuments(SearchResults<Hotel>  
searchResults)  
{  
    foreach ( SearchResult<Hotel> result in  
searchResults.GetResults())  
    {  
        Console.WriteLine(result.Document);  
    }  
  
    Console.WriteLine();  
}  
  
private static void WriteDocuments(AutocompleteResults autoResults)  
{  
    foreach ( AutocompleteItem result in autoResults.Results)  
    {  
        Console.WriteLine(result.Text);  
    }  
  
    Console.WriteLine();  
}
```

2. 쿼리를 실행하고 결과를 반환하는 RunQueries 메서드를 만듭니다. 결과는 Hotel 개체입니다. 이 샘플에서는 메서드 서명과 첫 번째 쿼리를 보여줍니다. 이 쿼리는 문서에서 선택한 필드를 사용하여 결과를 작성할 수 있도록 하는 Select 매개 변수를 보여줍니다.

```
C#  
  
// Run queries, use WriteDocuments to print output  
private static void RunQueries(SearchClient srchclient)  
{  
    SearchOptions options;  
    SearchResults<Hotel> response;  
  
    // Query 1  
    Console.WriteLine("Query #1: Search on empty term '*' to return  
all documents, showing a subset of fields...\n");  
  
    options = new SearchOptions()  
    {  
        IncludeTotalCount = true,  
        Filter = "",  
        OrderBy = { "" }  
    }
```

```

};

options.Select.Add("HotelId");
options.Select.Add("HotelName");
options.Select.Add("Address/City");

response = srchclient.Search<Hotel>("*", options);
WriteDocuments(response);

```

3. 두 번째 쿼리에서 용어를 검색하고 등급 0/4보다 큰 문서를 선택하는 필터를 추가한 다음 등급별로 내림차순으로 정렬합니다. 필터는 인덱스의 `IsFilterable` 필드를 통해 평가되는 부울 식입니다. 필터는 포함 또는 제외 값을 쿼리합니다. 따라서 필터 쿼리와 관련된 관련성 점수가 없습니다.

C#

```

// Query 2
Console.WriteLine("Query #2: Search on 'hotels', filter on 'Rating gt 4', sort by Rating in descending order...\n");

options = new SearchOptions()
{
    Filter = "Rating gt 4",
    OrderBy = { "Rating desc" }
};

options.Select.Add("HotelId");
options.Select.Add("HotelName");
options.Select.Add("Rating");

response = srchclient.Search<Hotel>("hotels", options);
WriteDocuments(response);

```

4. 세 번째 쿼리는 전체 텍스트 검색 작업의 범위를 특정 필드로 지정하는 데 사용되는 `searchFields`를 보여 줍니다.

C#

```

// Query 3
Console.WriteLine("Query #3: Limit search to specific fields (pool in Tags field)...\\n");

options = new SearchOptions()
{
    SearchFields = { "Tags" }
};

options.Select.Add("HotelId");
options.Select.Add("HotelName");
options.Select.Add("Tags");

```

```
response = srchclient.Search<Hotel>("pool", options);
WriteDocuments(response);
```

5. 네 번째 쿼리는 `facets` 패싯 탐색 구조를 구성하는 데 사용할 수 있는 방법을 보여 줍니다.

C#

```
// Query 4
Console.WriteLine("Query #4: Facet on 'Category'...\n");

options = new SearchOptions()
{
    Filter = ""
};

options.Facets.Add("Category");

options.Select.Add("HotelId");
options.Select.Add("HotelName");
options.Select.Add("Category");

response = srchclient.Search<Hotel>("*", options);
WriteDocuments(response);
```

6. 다섯 번째 쿼리에서 특정 문서를 반환합니다. 문서 조회는 결과 집합의 이벤트에 대한 일반적인 응답 `OnClick` 입니다.

C#

```
// Query 5
Console.WriteLine("Query #5: Look up a specific document...\\n");

Response<Hotel> lookupResponse;
lookupResponse = srchclient.GetDocument<Hotel>("3");

Console.WriteLine(lookupResponse.Value.HotelId);
```

7. 마지막 쿼리는 자동 완성 구문을 보여 줍니다. 이 구문은 인덱스에 정의한 제안기와 연결된 `sourceFields`에서 가능한 두 일치 항목으로 확인되는 `sa`의 부분 사용자 입력을 시뮬레이트합니다.

C#

```
// Query 6
Console.WriteLine("Query #6: Call Autocomplete on HotelName that
starts with 'sa'...\\n");
```

```
var autoresponse = srchclient.AutoComplete("sa", "sg");
WriteDocuments(autoresponse);
```

8. RunQueries 를 Main() 에 추가합니다.

C#

```
// Call the RunQueries method to invoke a series of queries
Console.WriteLine("Starting queries...\n");
RunQueries(srchclient);

// End the program
Console.WriteLine("{0}", "Complete. Press any key to end this
program...\n");
Console.ReadKey();
```

이전 쿼리는 전체 텍스트 검색, 필터 및 자동 완성과 같은 쿼리에서 용어를 일치시키는 여러 방법을 보여줍니다.

전체 텍스트 검색 및 필터는 `SearchClient.Search` 메서드를 사용하여 수행됩니다. 검색 쿼리는 `searchText` 문자열로 전달할 수 있는 반면, 필터 식은 `SearchOptions` 클래스의 `Filter` 속성으로 전달할 수 있습니다. 검색하지 않고 필터링하려면 `Search` 메서드의 `searchText` 매개 변수에 대한 `"*"`을 전달합니다. 필터링하지 않고 검색하려면 `Filter` 속성을 설정하지 않고 그대로 두거나 `SearchOptions` 인스턴스에 전달하지 않아야 합니다.

## 프로그램 실행

F5 키를 눌러 앱을 다시 빌드하고 프로그램을 완전히 실행합니다.

출력에는 쿼리 정보 및 결과가 추가된 `Console.WriteLine`의 메시지가 포함됩니다.

## 리소스 정리

본인 소유의 구독으로 이 모듈을 진행하고 있는 경우에는 프로젝트가 끝날 때 여기에서 만든 리소스가 계속 필요한지 확인하는 것이 좋습니다. 계속 실행되는 리소스에는 요금이 부과될 수 있습니다. 리소스를 개별적으로 삭제하거나 리소스 그룹을 삭제하여 전체 리소스 세트를 삭제할 수 있습니다.

왼쪽 탐색 창의 모든 리소스 또는 리소스 그룹 링크를 사용하여 포털에서 리소스를 찾고 관리할 수 있습니다.

무료 서비스를 사용하는 경우 인덱스, 인덱서, 데이터 원본 3개로 제한됩니다. 포털에서 개별 항목을 삭제하여 제한 이하로 유지할 수 있습니다.

## 다음 단계

이 빠른 시작에서는 인덱스를 만들고 문서와 함께 로드하며 쿼리를 실행하는 일련의 작업을 수행했습니다. 여러 단계에서, 가독성과 이해를 돋기 위해 손쉬운 방법을 사용하여 코드를 간소화했습니다. 이제 기본 개념에 익숙해졌으므로 웹앱에서 Azure AI 검색 API를 호출하는 자습서를 시도해 보세요.

[자습서: 웹앱에 검색 추가](#)

## 피드백

이 페이지가 도움이 되었나요?

 Yes

 No

[제품 사용자 의견 제공](#) | [Microsoft Q&A에서 도움말 보기](#)

# 빠른 시작: .NET 또는 Python을 사용한 의미 체계 순위

아티클 • 2024. 10. 22.

Azure AI 검색에서 [의미 체계 순위](#)는 Microsoft의 기계독해를 사용하여 검색 결과의 점수를 다시 매기고 가장 의미 관련성이 높은 일치 항목을 목록 맨 위에 표시하는 쿼리 쪽 기능입니다. 콘텐츠와 쿼리에 따라 의미 체계 순위는 개발자의 작업을 최소화하면서 [검색 관련성을 대폭 향상](#)할 수 있습니다.

이 빠른 시작에서는 의미 순위 매기기를 호출하는 인덱스 및 쿼리 수정 사항을 안내합니다.

## ① 참고

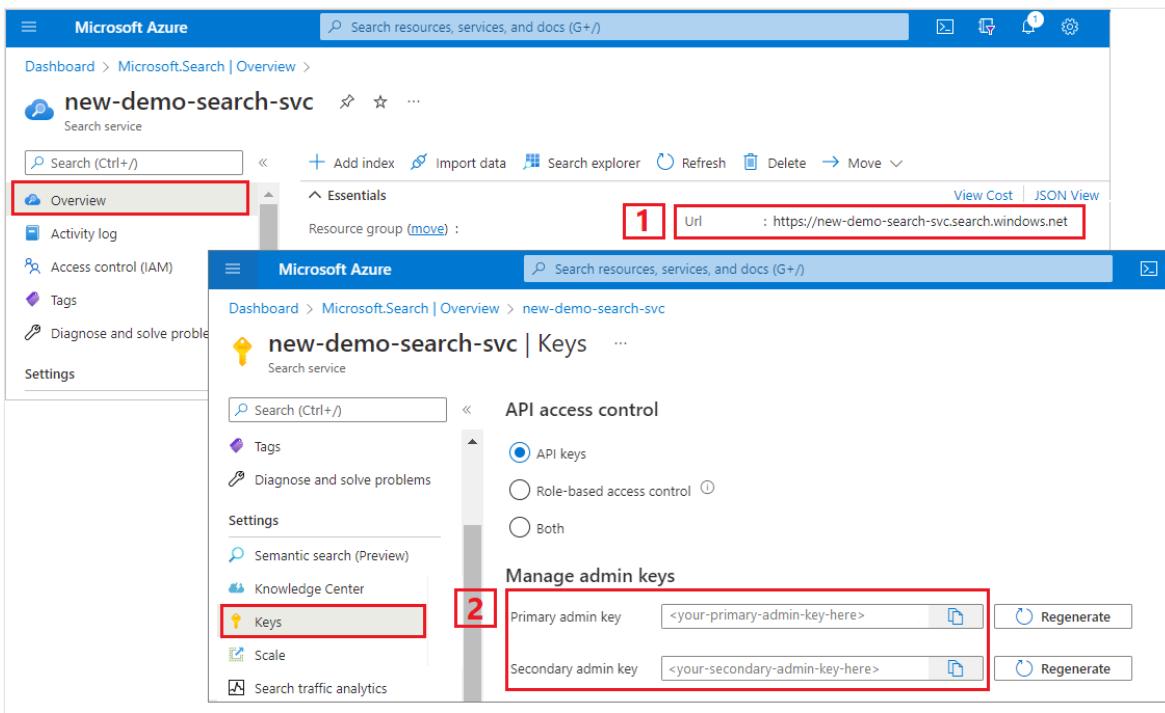
ChatGPT 상호 작용을 사용하는 Azure AI Search 솔루션 예제는 이 [데모 또는 이 가속기를 참조](#)하세요.

## 필수 조건

- 활성 구독이 있는 Azure 계정. 무료로 [계정을 만들 수 있습니다](#).
- 의미 체계 순위가 설정된 [기본 계층 이상의](#) Azure AI Search 리소스입니다.
- API 키 및 검색 서비스 엔드포인트. [Azure Portal](#)에 로그인하고 [검색 서비스를 찾습니다](#).

개요에서 URL을 복사하고 이후 단계를 위해 저장합니다. 엔드포인트의 예는 다음과 같습니다. <https://mydemo.search.windows.net>

키에서 개체를 만들고 삭제할 수 있는 모든 권한의 관리자 키를 복사하고 저장합니다. 상호 교환 가능한 기본 키와 보조 키가 있습니다. 둘 중 하나를 선택합니다.



## 의미 체계 순위 추가

의미 순위매기기를 사용하려면 검색 인덱스에 의미 체계 구성을 추가하고 쿼리에 매개 변수를 추가합니다. 기존 인덱스가 있는 경우 검색 가능한 콘텐츠의 구조에 영향을 주지 않으므로 콘텐츠를 다시 인덱싱하지 않고도 이러한 변경을 수행할 수 있습니다.

- 의미 체계 구성은 의미 체계 순위 다시 지정에 사용되는 제목, 키워드 및 콘텐츠에 기여하는 필드에 대한 우선 순위 순서를 설정합니다. 필드 우선 순위를 지정하면 처리 속도가 빨라집니다.
- 의미 순위매기기를 호출하는 쿼리에는 쿼리 형식, 캡션 및 답변이 반환되는지 여부에 대한 매개 변수가 포함됩니다. 이러한 매개 변수를 기존 쿼리 논리에 추가할 수 있습니다. 다른 매개 변수와는 충돌하지 않습니다.

.NET

Azure.Search.Documents 클라이언트 라이브러리를 사용하여 기존 검색 인덱스로 의미 체계 순위를 추가하여 콘솔 애플리케이션을 빌드합니다.

또는 소스 코드를 다운로드하여 완료된 프로젝트로 시작할 수 있습니다.

## 환경 설정

- Visual Studio를 시작하고 콘솔 앱용 새 프로젝트를 만듭니다.
- 도구>NuGet 패키지 관리자에서 솔루션의 NuGet 패키지 관리...를 선택합니다.

3. 찾아보기를 선택합니다.
4. Azure.Search.Documents 패키지를 검색하고 안정적인 최신 버전을 선택합니다.
5. 설치를 선택하여 프로젝트 및 솔루션에 어셈블리를 추가합니다.

## 검색 클라이언트 만들기

1. Program.cs에서 다음 `using` 지시문을 추가합니다.

```
C#
using Azure;
using Azure.Search.Documents;
using Azure.Search.Documents.Indexes;
using Azure.Search.Documents.Indexes.Models;
using Azure.Search.Documents.Models;
```

2. 두 개의 클라이언트를 만듭니다. `SearchIndexClient`는 인덱스를 만들고, `SearchClient`는 기존 인덱스를 로드하고 쿼리합니다.

두 클라이언트 모두 만들기/삭제 권한으로 인증을 위해 서비스 엔드포인트와 관리자 API 키가 필요합니다. 그러나 코드는 URI를 작성하므로 속성의 검색 서비스 이름 `serviceName`만 지정합니다. 포함 `https://` 안 함 또는 `.search.windows.net.`

```
C#
static void Main(string[] args)
{
    string serviceName = "<YOUR-SEARCH-SERVICE-NAME>";
    string apiKey = "<YOUR-SEARCH-ADMIN-API-KEY>";
    string indexName = "hotels-quickstart";

    // Create a SearchIndexClient to send create/delete index
    commands
    Uri serviceEndpoint = new
    Uri($"https://'{serviceName}'.search.windows.net/");
    AzureKeyCredential credential = new
    AzureKeyCredential(apiKey);
    SearchIndexClient adminClient = new
    SearchIndexClient(serviceEndpoint, credential);

    // Create a SearchClient to load and query documents
    SearchClient srchclient = new SearchClient(serviceEndpoint,
    indexName, credential);
```

```
    . . .  
}
```

## 인덱스 만들기

`SemanticConfiguration`을 포함하도록 인덱스 스키마를 만들거나 업데이트합니다. 기존 인덱스를 업데이트하는 경우 문서 구조가 변경되지 않기 때문에 이 설정을 다시 인덱싱할 필요가 없습니다.

C#

```
// Create hotels-quickstart index  
private static void CreateIndex(string indexName, SearchIndexClient  
adminClient)  
{  
  
    FieldBuilder fieldBuilder = new FieldBuilder();  
    var searchFields = fieldBuilder.Build(typeof(Hotel));  
  
    var definition = new SearchIndex(indexName, searchFields);  
    var suggester = new SearchSuggester("sg", new[] { "HotelName",  
"Category", "Address/City", "Address/StateProvince" });  
    definition.Suggesters.Add(suggester);  
    definition.SemanticSearch = new SemanticSearch  
    {  
        Configurations =  
        {  
            new SemanticConfiguration("my-semantic-config", new()  
            {  
                TitleField = new SemanticField("HotelName"),  
                ContentFields =  
                {  
                    new SemanticField("Description"),  
                    new SemanticField("Description_fr")  
                },  
                KeywordsFields =  
                {  
                    new SemanticField("Tags"),  
                    new SemanticField("Category")  
                }  
            })  
        }  
    };  
  
    adminClient.CreateOrUpdateIndex(definition);  
}
```

다음 코드는 검색 서비스에 인덱스를 만듭니다.

C#

```
// Create index
Console.WriteLine("{0}", "Creating index...\n");
CreateIndex(indexName, adminClient);

SearchClient ingestorClient = adminClient.GetSearchClient(indexName);
```

## 문서 로드

Azure AI 검색은 서비스에 저장된 콘텐츠를 검색합니다. 문서를 업로드하는 코드는 [전체 텍스트 검색을 위한 C# 빠른 시작](#)과 동일하므로 여기에서 중복할 필요가 없습니다. 이름, 주소 및 설명이 있는 4개의 호텔이 있어야 합니다. 솔루션에는 호텔 및 주소 유형이 있어야 합니다.

## 인덱스 검색

매개 변수를 지정하기 위한 검색 옵션을 사용하여 의미 순위매기기를 호출하는 쿼리는 다음과 같습니다.

C#

```
Console.WriteLine("Example of a semantic query.");

options = new SearchOptions()
{
    QueryType = Azure.Search.Documents.Models.SearchQueryType.Semantic,
    SemanticSearch = new()
    {
        SemanticConfigurationName = "my-semantic-config",
        QueryCaption = new(QueryCaptionType.Extractive)
    }
};
options.Select.Add("HotelName");
options.Select.Add("Category");
options.Select.Add("Description");

// response = srchclient.Search<Hotel>("*", options);
response = srchclient.Search<Hotel>("what hotel has a good restaurant on
site", options);
WriteDocuments(response);
```

비교를 위해 용어 빈도 및 근접성을 기반으로 기본 BM25 순위를 사용하는 쿼리의 결과는 다음과 같습니다. BM25 순위 알고리즘은 "어떤 호텔 구내에 좋은 레스토랑이 있는가?" 쿼리를 감안할 때 다음 스크린샷에 표시된 순서대로 일치 항목을 반환합니다.

```
C:\test\semantic-search-quick X + ^ - □ X

Query #2: Full text search on 'what [hotel] has a good [restaurant] on [site]'
with BM25 ranking...

Name: Sublime Cliff Hotel
Description: Sublime Cliff [Hotel] is located in the heart of the historic
center of Sublime in an extremely vibrant and lively area within short wa-
lking distance to the [sites] and landmarks of the city and is surrounded b-
y the extraordinary beauty of churches, buildings, shops and monuments. S-
ublime Cliff is part of a lovingly restored 1800 palace.

Name: Twin Dome Motel
Description: The [hotel] is situated in a nineteenth century plaza, which
has been expanded and renovated to the highest architectural standards to
create a modern, functional and first-class [hotel] in which art and uniqu-
e historical elements coexist with the most modern comforts.

Name: Triple Landscape Hotel
Description: The [Hotel] stands out for its gastronomic excellence under th-
e management of William Dough, who advises on and oversees all of the Hot-
el's [restaurant] services.

Name: Secret Point Motel
Description: The [hotel] is ideally located on the main commercial artery o-
f the city in the heart of New York. A few minutes away is Time's Square
and the historic centre of the city, as well as other places of interest
that make New York one of America's most attractive and cosmopolitan citi-
es.
```

반면, 의미 체계 순위가 동일한 쿼리("어떤 호텔 구내에 좋은 레스토랑이 있는가?")에 적용되는 경우 쿼리에 대한 의미 체계 관련성에 따라 결과 순위가 다시 지정됩니다. 여기서 가장 좋은 결과는 레스토랑이 있는 호텔이며, 이는 사용자의 기대에 더욱 잘 부합합니다.

```
C:\test\semantic-search-quick X + - □ ×
```

Query #3: Invoke semantic search on the same query..

Name: Triple Landscape Hotel  
Description: The Hotel stands out for its gastronomic excellence under the management of William Dough, who advises on and oversees all of the Hotel's restaurant services.

Name: Sublime Cliff Hotel  
Description: Sublime Cliff Hotel is located in the heart of the historic center of Sublime in an extremely vibrant and lively area within short walking distance to the sites and landmarks of the city and is surrounded by the extraordinary beauty of churches, buildings, shops and monuments. Sublime Cliff is part of a lovingly restored 1800 palace.

Name: Secret Point Motel  
Description: The hotel is ideally located on the main commercial artery of the city in the heart of New York. A few minutes away is Time's Square and the historic centre of the city, as well as other places of interest that make New York one of America's most attractive and cosmopolitan cities.

Name: Twin Dome Motel  
Description: The hotel is situated in a nineteenth century plaza, which has been expanded and renovated to the highest architectural standards to create a modern, functional and first-class hotel in which art and unique historical elements coexist with the most modern comforts.

## 프로그램 실행

F5 키를 눌러서 애플리케이션을 다시 빌드하고 프로그램 전체를 실행합니다.

출력에는 쿼리 정보 및 결과가 추가된 [Console.WriteLine](#)의 메시지가 포함됩니다.

## 리소스 정리

본인 소유의 구독으로 이 모듈을 진행하고 있는 경우에는 프로젝트가 끝날 때 여기에서 만든 리소스가 계속 필요한지 확인하는 것이 좋습니다. 계속 실행되는 리소스에는 요금이 부과될 수 있습니다. 리소스를 개별적으로 삭제하거나 리소스 그룹을 삭제하여 전체 리소스 세트를 삭제할 수 있습니다.

왼쪽 탐색 창의 **모든 리소스** 또는 **리소스 그룹** 링크를 사용하여 포털에서 리소스를 찾고 관리할 수 있습니다.

## 다음 단계

이 빠른 시작에서는 기존 인덱스의 의미 체계 순위를 호출하는 방법을 알아보았습니다. 다음 단계로 사용자 고유의 인덱스에 대한 의미 체계 순위를 시도하는 것이 좋습니다. 그러나 데모를 계속하려면 다음 링크를 방문하세요.

자습서: 웹앱에 검색 추가

## 피드백

이 페이지가 도움이 되었나요?

👍 Yes

👎 No

제품 사용자 의견 제공 | Microsoft Q&A에서 도움말 보기

# 빠른 시작: 사용자 고유의 데이터를 사용하여 Azure OpenAI 모델과 채팅

아티클 · 2024. 10. 22.

이 빠른 시작에서는 Azure OpenAI 모델에서 사용자 고유의 데이터를 사용할 수 있습니다. 데이터에 Azure OpenAI의 모델을 사용하면 더 빠르고 정확한 커뮤니케이션을 가능하게 하는 강력한 대화형 AI 플랫폼을 제공할 수 있습니다.

## 필수 조건

다음 리소스:

- [Azure OpenAI](#)
- [Azure Blob Storage](#)
- [Azure AI 검색](#)
- [지원되는 모델을 사용하여 지원되는 지역에 배포된 Azure OpenAI 리소스](#).
  - 적어도 Azure OpenAI 리소스에 대한 [Cognitive Services](#) 기여자 역할이 할당되어야 합니다.
- 자체 데이터가 없으면 [GitHub](#)에서 데이터 예를 다운로드합니다.

필수 조건에 문제가 있습니다.

## Azure OpenAI Studio를 사용하여 데이터 추가

### 💡 팁

[Azure 개발자 CLI](#)를 사용하여 Azure OpenAI On Your Data에 필요한 리소스를 프로그래밍 방식으로 만들 수 있음

Azure OpenAI Studio로 이동한 다음, Azure OpenAI 리소스에 액세스할 수 있는 자격 증명으로 로그인합니다. 로그인 워크플로 도중 또는 이후에 적절한 디렉터리, Azure 구독 및 Azure OpenAI 리소스를 선택합니다.

- 자체 데이터 가져오기 타일을 선택합니다.

The screenshot shows the Azure OpenAI Studio interface. On the left is a sidebar with navigation links: Home, Get started, Model catalog, Playgrounds (Chat, Assistants, Images, Completions), Tools (Fine-tuning, Batch jobs), Shared resources (Deployments, Quota, Content filters, Data files, Vector stores). The main content area has a title 'Welcome to Azure OpenAI service' and a sub-section 'Get started'. It lists four playgrounds: 'Assistants playground', 'Chat playground', 'Completions playground', and 'Images playground', each with a 'Try it now' button. To the right, there's a section titled 'Bring your own data' with a 'Try it now' button, and a circular icon with a magnifying glass.

2. 채팅 플레이그라운드에서 데이터 추가를 선택한 다음 데이터 원본 추가를 선택합니다.

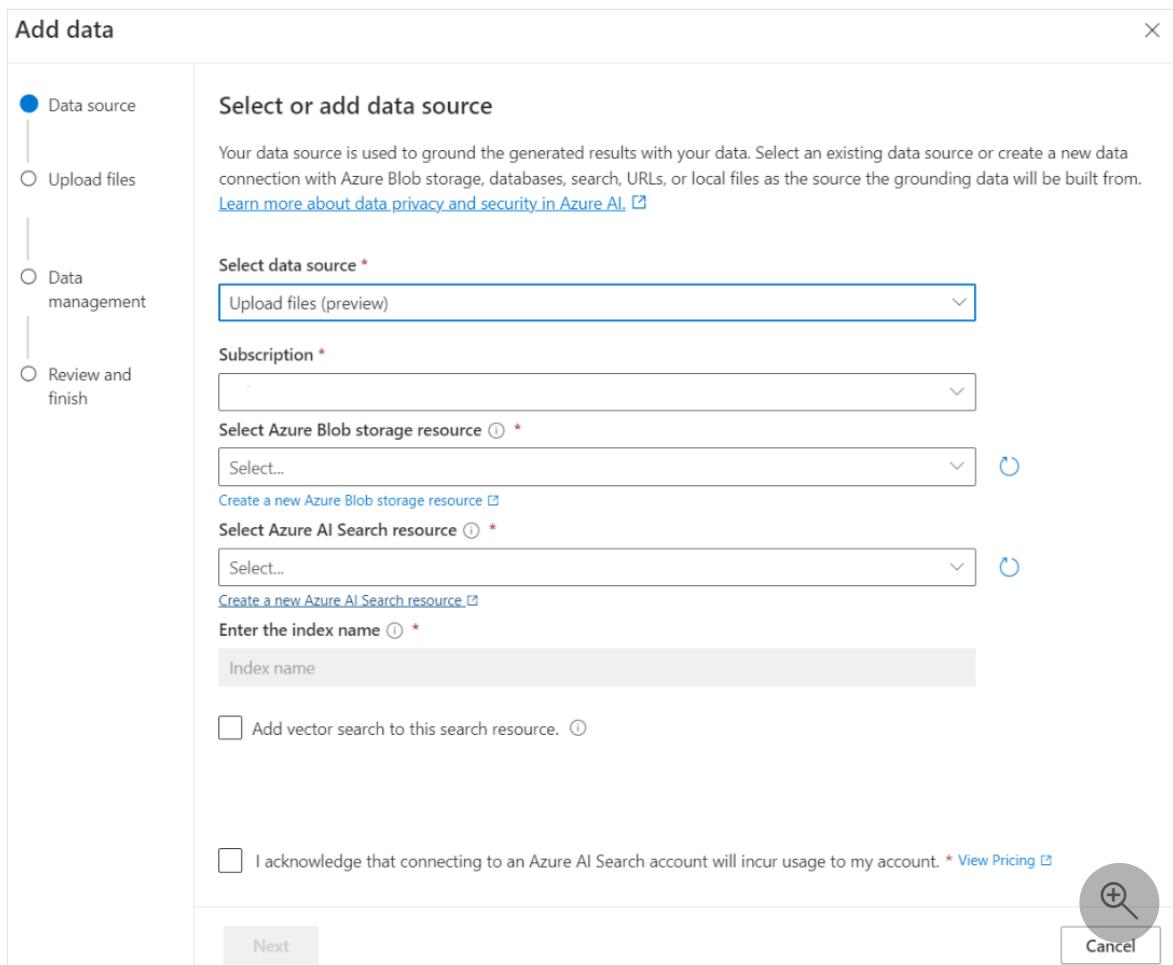
The screenshot shows the 'Chat playground' setup screen. The sidebar on the left is identical to the previous one. The main area has a title 'Chat playground' and a toolbar with 'Apply changes', 'View Code', 'Deploy', 'Import', 'Export', and 'Prompt' buttons. The 'Setup' section contains a 'Deployment' dropdown set to 'gpt-4o (version:2024-05-13)'. Below it are 'System message' and 'Parameters' fields. A large red box highlights the 'Add your data' button. Another red box highlights the '+ Add a data source' button. On the right, there's a sidebar with a robot icon and text about starting a chat, and a bottom input field with placeholder text 'Type user query here. (Shift + Enter for multiple lines)'.

3. 표시되는 창의 데이터 원본 선택에서 파일 업로드(미리 보기)를 선택합니다. Azure OpenAI는 데이터에 액세스하고 인덱싱하기 위해 스토리지 리소스와 검색 리소스가 모두 필요합니다.

## 💡 팁

- 자세한 내용은 다음 리소스를 참조하세요.
  - [데이터 원본 옵션](#)
  - [지원되는 파일 유형 및 형식](#)
- 긴 텍스트가 있는 문서 및 데이터 세트의 경우 사용 가능한 [데이터 준비 스크립트](#) 를 사용하는 것이 좋습니다.

- a. Azure OpenAI가 스토리지 계정에 액세스하려면 [CORS\(원본 간 리소스 공유\)](#) 를 설정해야 합니다. Azure Blob Storage 리소스에 대해 CORS가 아직 켜져 있지 않은 경우 [CORS 켜기](#)를 선택합니다.
- b. Azure AI 검색 리소스를 선택하고, 연결하면 계정에서 사용량이 발생한다는 데에 확인을 선택합니다. 그런 후 [다음](#)을 선택합니다.



4. 파일 업로드 창에서 파일 찾아보기를 선택하고 필수 조건 섹션에서 다운로드한 파일이나 자체 데이터를 선택합니다. 파일 업로드를 선택합니다. 그런 후 다음을 선택합니다.
5. 데이터 관리 창에서 인덱스에 의미 체계 검색 또는 벡터 검색을 사용할지를 선택할 수 있습니다.

## ① 중요

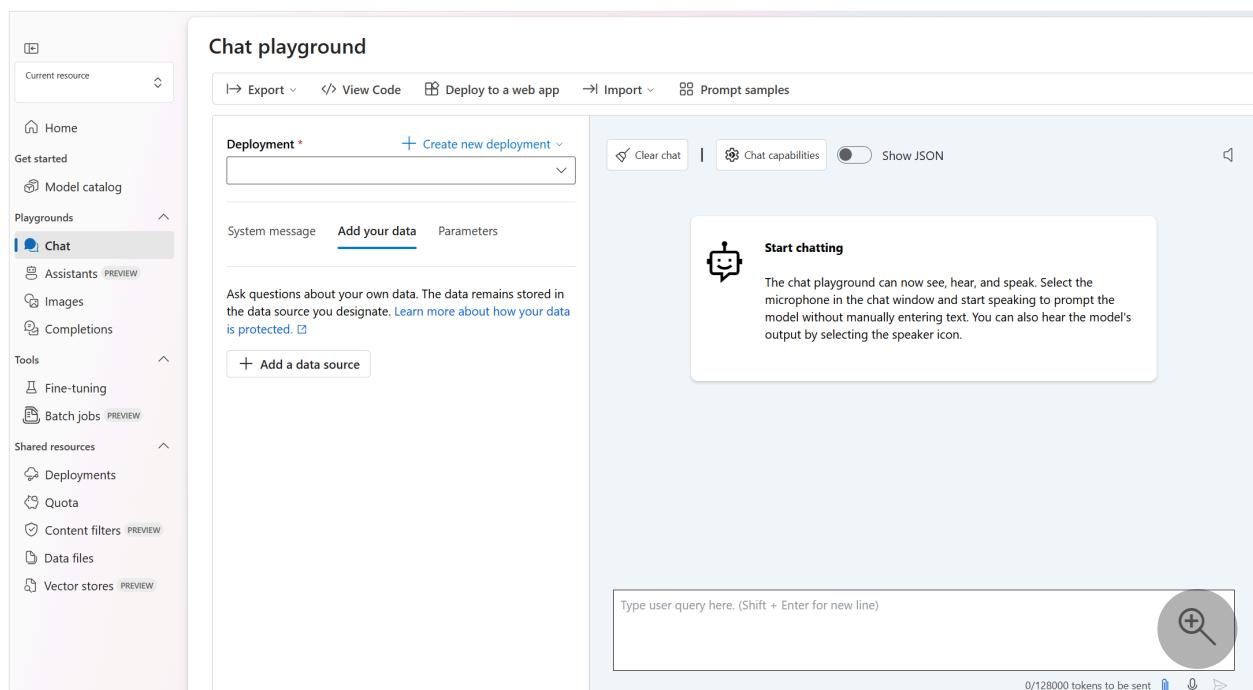
- 의미 체계 검색 및 벡터 검색 에는 추가 가격 책정이 적용됩니다. 의미 체계 검색 또는 벡터 검색을 사용하도록 설정하려면 **기본 이상의 SKU**를 선택해야 합니다. 자세한 내용은 [가격 책정 계층 차이](#) 및 [서비스 제한](#)을 참조하세요.
- 정보 검색 및 모델 응답의 품질을 향상하려면 영어, 프랑스어, 스페인어, 포르투갈어, 이탈리아어, 독일, 중국어(Zh), 일본어, 한국어, 러시아어 및 아랍어와 같은 데이터 원본 언어에 대해 의미 체계 검색을 사용하도록 설정하는 것이 좋습니다.

6. 입력한 세부 정보를 검토하고 저장 및 닫기를 선택하세요. 이제 모델과 채팅할 수 있으며 모델은 당신의 데이터 정보를 사용하여 응답을 생성할 것입니다.

내 데이터를 추가하는 데 문제가 발생했습니다.

## 채팅 플레이그라운드

채팅 플레이그라운드를 통해 코드 없는 접근 방식으로 Azure OpenAI 기능 탐색을 시작합니다. 플레이그라운드는 완료를 생성하는 프롬프트를 제출할 수 있는 간단한 텍스트 상자입니다. 이 페이지에서 쉽게 기능을 반복하고 실험해 볼 수 있습니다.



플레이그라운드에서는 채팅 환경을 맞춤화할 수 있는 옵션을 제공합니다. 오른쪽에서 배포를 선택하면 인덱스의 검색 결과를 사용하여 응답을 생성하는 모델을 결정할 수 있습니다. 향후 생성되는 응답에 대한 대화 기록으로 포함할 과거 메시지 수를 선택합니다. 대

화 기록은 관련 응답을 생성하기 위한 컨텍스트를 제공하지만 토큰 사용량도 소비합니다. 입력 토큰 진행률 표시기는 제출한 질문의 토큰 수를 추적합니다.

왼쪽에 있는 고급 설정은 데이터 검색을 제어하고 관련 정보를 검색할 수 있는 런타임 매개 변수입니다. 좋은 사용 사례는 데이터를 기반으로만 응답이 생성되도록 하거나 모델이 데이터에 존재하는 정보를 기반으로 응답을 생성할 수 없는 경우입니다.

- **엄격성**은 유사성 점수를 기반으로 검색 문서를 필터링하는 시스템의 공격성을 결정합니다. 엄격도를 5로 설정하면 시스템이 매우 높은 유사성 임계값을 적용하여 문서를 적극적으로 필터링한다는 의미입니다. 의미 체계 검색은 순위 모델이 쿼리의 의도를 더욱 효과적으로 유추하기 때문에 이 시나리오에서는 의미 체계 순위가 유용할 수 있습니다. 엄격성 수준이 낮을수록 자세한 답변이 생성되지만 인덱스에 없는 정보가 포함될 수도 있습니다. 기본적으로 3으로 설정되어 있습니다.
- **검색된 문서**는 3, 5, 10 또는 20으로 설정할 수 있는 정수이며 최종 응답을 수식화하기 위해 대규모 언어 모델에 제공되는 문서 청크 수를 제어합니다. 기본적으로 5로 설정됩니다.
- **데이터에 대한 응답 제한**이 사용하도록 설정되면 모델은 응답을 위해 문서에만 의존하려고 시도합니다. 이는 기본적으로 true로 설정됩니다.

Assistant setup ×

Prompt Add your data

Gain insights into your own data source. Your data is stored securely in your Azure subscription. [Learn more about how your data is protected.](#)

Data source:	Search Resource:
Upload Files	demo
Index:	Chunk Size:
.	1024

**Advanced settings** ▼

Limit responses to your data content  ⓘ

Strictness (1-5)  ⓘ

4

Retrieved documents (3-20)  ⓘ

20

Remove data source

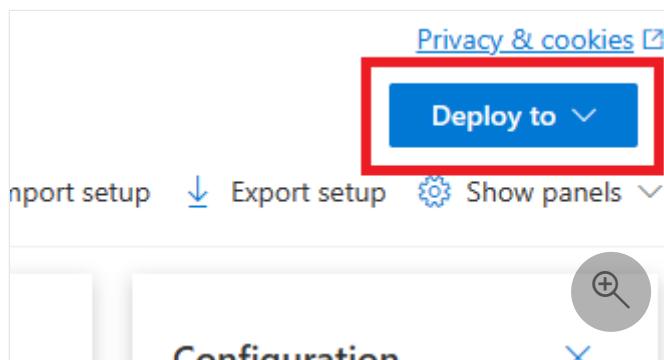
첫 번째 쿼리를 전송합니다. 채팅 모델은 질문 및 답변 연습에서 가장 효과적으로 수행됩니다. 예를 들어, "사용 가능한 건강 보험은 무엇인가요?" 또는 "건강 플러스 옵션은 무엇인가요?"입니다.

"가장 자주 사용되는 건강 보험은 무엇인가요?"와 같이 데이터 분석이 필요한 쿼리는 실패할 수 있습니다. "내가 업로드한 문서 수는 몇 개인가요?"와 같이 모든 데이터에 대한 정보가 필요한 쿼리도 실패할 가능성이 높습니다. 검색 엔진은 쿼리에 정확한 용어나 유사한 용어, 구 또는 구성이 있는 청크를 찾는다는 점을 기억하세요. 모델이 질문을 이해할 수도 있지만 검색 결과가 데이터 세트의 일부인 경우 해당 질문에 답하는 데 적합한 정보가 아닙니다.

채팅은 응답에서 반환된 문서(청크) 수에 따라 제한됩니다(Azure OpenAI Studio 플레이 그라운드의 경우 3~20개로 제한됨). 여러분이 상상할 수 있듯이 "모든 제목"에 대한 질문을 제기하려면 전체 벡터 저장소의 전체 검사가 필요합니다.

## 모델 배포

Azure OpenAI Studio의 환경에 만족하면 **배포 대상** 단추를 선택하여 스튜디오에서 직접 웹앱을 배포할 수 있습니다.



이를 통해 모델에서 자체 데이터를 사용하는 경우 독립형 웹 애플리케이션에 배포하거나 Copilot Studio(미리 보기)의 Copilot에 배포할 수 있는 옵션이 제공됩니다.

예를 들어 웹앱을 배포하도록 선택하는 경우:

웹앱을 처음 배포할 때 새 웹앱 만들기를 선택해야 합니다. 앱 URL의 일부가 될 앱의 이름을 선택합니다. 예: `https://<appname>.azurewebsites.net.`

게시된 앱에 대한 구독, 리소스 그룹, 위치 및 가격 책정 계획을 선택합니다. 기존 앱을 업데이트하려면 기존 웹앱에 게시를 선택하고 드롭다운 메뉴에서 이전 앱의 이름을 선택합니다.

웹앱을 배포하도록 선택하는 경우 웹앱을 사용하기 위한 [중요한 고려 사항](#)을 참조하세요.

**모델 배포와 관련된 문제가 발생했습니다.**

# 리소스 정리

Azure OpenAI 또는 Azure AI 검색 리소스를 정리하고 제거하려면 리소스 또는 리소스 그룹을 삭제하면 됩니다. 리소스 그룹을 삭제하면 해당 리소스 그룹에 연결된 다른 모든 리소스가 함께 삭제됩니다.

- [Azure AI 서비스 리소스](#)
- [Azure AI 검색 리소스](#)
- [Azure App Service 리소스](#)

## 다음 단계

- [Azure OpenAI Service에서 데이터 사용에 대해 자세히 알아보기](#)
- [Github에서 채팅 앱 샘플 코드 보기](#).

---

## 피드백

이 페이지가 도움이 되었나요?

 Yes

 No

[제품 사용자 의견 제공](#) | [Microsoft Q&A에서 도움말 보기](#)

# 빠른 시작: Azure Portal에서 검색 인덱스 만들기

아티클 • 2024. 10. 16.

이 Azure AI 검색 빠른 시작에서는 [데이터 가져오기 마법사](#)와 Microsoft가 호스트하는 가상의 호텔 데이터로 구성된 기본 제공 샘플 데이터 원본을 사용하여 첫 번째 검색 인덱스를 만듭니다. 마법사는 코드 없이 검색 인덱스를 만드는 과정을 안내하여 몇 분 내에 흥미로운 쿼리를 작성하는 데 도움이 됩니다.

마법사는 검색 서비스에 여러 개체(검색 가능한 인덱스)를 생성할 뿐만 아니라 자동화된 데이터 검색을 위한 인덱서 및 데이터 원본 연결도 생성합니다. 이 빠른 시작이 끝나면 각 개체를 검토합니다.

## ① 참고

데이터 가져오기 마법사에는 이 빠른 시작에서 다루지 않은 OCR, 텍스트 번역 및 기타 AI 보강 옵션이 포함되어 있습니다. 적용 AI에 포커스를 맞춘 유사한 연습은 [빠른 시작: Azure Portal에서 기술 세트 만들기](#)를 참조하세요.

## 필수 조건

- 활성 구독이 있는 Azure 계정. [체험 계정을 만듭니다](#).
- Azure AI 검색 서비스(임의 계층 및 임의 지역) [서비스를 만들거나](#) 현재 구독에서 [기존 서비스를 찾습니다](#). 이 빠른 시작에서는 체험 서비스를 사용할 수 있습니다.

기본 제공 샘플 데이터를 사용하는 이 빠른 시작의 경우, 검색 서비스에 [네트워크 액세스 제어](#)가 없는지 확인합니다. 포털 컨트롤러는 퍼블릭 엔드포인트를 사용하여 Microsoft에서 호스팅하는 기본 제공 샘플 데이터 원본에서 데이터와 메타데이터를 검색합니다. 자세한 정보는 [가져오기 마법사의 보안 연결](#)을 참조하세요.

## 공간 확인

많은 고객이 무료 서비스를 시작합니다. 무료 계층은 3개의 인덱스, 3개의 데이터 소스 및 3개의 인덱서로 제한됩니다. 시작하기 전에 추가 항목에 대한 공간이 있는지 확인합니다. 이 빠른 시작에서는 각 개체를 하나씩 만듭니다.

이미 보유한 인덱스, 인덱서 및 데이터 원본의 수를 확인하려면 해당 서비스의 [개요 > 사용량](#) 탭을 확인합니다.

The screenshot shows the Azure portal interface for a search service. The left sidebar has sections like Home, Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Search management (Indexes, Indexers, Data sources, Aliases, Skillsets, Debug sessions), Settings (Semantic ranker, Knowledge Center, Keys), and Copilot. The main content area has tabs for Overview, Essentials, Get started, Properties, Usage (which is selected and highlighted with a red box), and Monitoring. Below these are five circular charts: Storage (24.6%, Current 12.31 MB, Quota 50 MB), Vector index size (0 Bytes, Quota 0 Bytes), Indexes (66.7%, Current 2, Quota 3), Indexers (66.7%, Current 2, Quota 3), and Data sources (66.7%, Current 2, Quota 3). A 'View indexes' link is visible under the Vector index size chart.

## 마법사 시작

1. Azure 계정으로 [Azure Portal](#)에 로그인하고 Azure AI 검색 서비스로 이동합니다.
2. **개요** 페이지에서 **데이터 가져오기**를 선택하여 마법사를 시작합니다.



## 인덱스 만들기 및 로드

이 섹션에서는 4단계에 걸쳐 인덱스를 만들고 로드합니다.

### 데이터 원본에 연결

마법사는 Azure Cosmos DB에서 Microsoft가 호스트하는 샘플 데이터에 대한 데이터 원본 연결을 만듭니다. 이 샘플 데이터는 퍼블릭 엔드포인트를 통해 액세스되고 검색됩니다. 이 빠른 시작을 실행하기 위해 고유한 Azure Cosmos DB 계정 또는 원본 파일이 필요하지 않습니다.

1. 데이터에 연결에서 데이터 원본 드롭다운 목록을 확장하고 샘플을 선택합니다.
2. 기본 제공 샘플 목록에서 **hotels-sample**을 선택합니다.

Home &gt; free-demo-search-svc &gt;

## Import data

X

[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 AI 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 \(Optional\)](#)[Give feedback](#)

3. 다음: 인식 기술 추가(선택 사항)을 선택하여 계속 진행합니다.

## 인식 기술의 구성 건너뛰기

데이터 가져오기 마법사는 기술 세트 생성과 인덱싱에 대한 [AI 보강](#)을 지원합니다.

1. 이 빠른 시작에서는 **인식 기술 추가** 탭의 AI 보강 구성 옵션을 무시하세요.
2. **건너뛰기: 대상 인덱스 사용자 지정**을 선택하여 계속 진행합니다.

[Previous: Connect to your data](#)[Skip to: Customize target index](#)

### 💡 팁

AI 보강에 관심이 있으신가요? 이 [빠른 시작: Azure Portal에서 기술 세트 만들기](#)를 사용해 보세요.

## 인덱스 구성

마법사는 기본 제공 hotels-sample index에 대한 스키마를 유추합니다. 다음 단계에 따라 인덱스를 구성합니다.

1. **인덱스 이름(hotels-sample-index)** 및 **키 필드(HotellId)**에 대해 시스템에서 생성된 값을 허용합니다.

2. 모든 필드 특성에 대해 시스템에서 생성된 값을 허용합니다.

3. 다음: 인덱서 만들기를 선택하여 계속 진행합니다.

The screenshot shows the 'Create an indexer' wizard in the Azure portal. The 'Fields' section is displayed, listing various hotel properties. Each field's configuration includes its name, type, and several checkboxes for searchability (Retrievable, Filterable, Sortable, Facetable, Searchable) and analysis (Analyzer, Suggester). Some fields like 'HotelId' and 'Rating' have their 'Retrievable' and 'Filterable' checkboxes checked. Others like 'Address' and 'Location' have their 'Searchable' checkboxes checked. Analyzers like 'English - Micro...' and 'French - Micro...' are assigned to specific fields. The 'Next: Create an indexer' button is highlighted with a red border at the bottom of the form.

인덱스에는 최소한 **인덱스 이름** 및 **필드** 컬렉션이 필요합니다. 각 문서를 고유하게 식별하기 위해 하나의 필드를 문서 키로 표시해야 합니다. 값은 항상 문자열입니다. 마법사는 고유한 문자열 필드를 검색하고 키에 대해 하나를 선택합니다.

각 필드에는 검색 인덱스에서 필드를 사용하는 방법을 제어하는 이름, 데이터 형식 및 특성이 있습니다. 확인란에서 다음 특성을 사용하거나 사용하지 않도록 설정할 수 있습니다.

- **Retrievable:** 쿼리 응답에서 반환된 필드입니다.
- **Filterable:** 필터 식을 허용하는 필드입니다.
- **Sortable:** orderby 식을 허용하는 필드입니다.
- **Facetable:** 패싯 탐색 구조에 사용되는 필드입니다.
- **Searchable:** 전체 텍스트 검색에 사용되는 필드입니다. 문자열은 검색할 수 있습니다. 숫자 필드와 부울 필드는 종종 검색할 수 없다고 표시됩니다.

문자열은 **Retrievable** 및 **Searchable** 특성이 사용됩니다. 정수는 **Retrievable**, **Filterable**, **Sortable** 및 **Facetable** 특성이 사용됩니다.

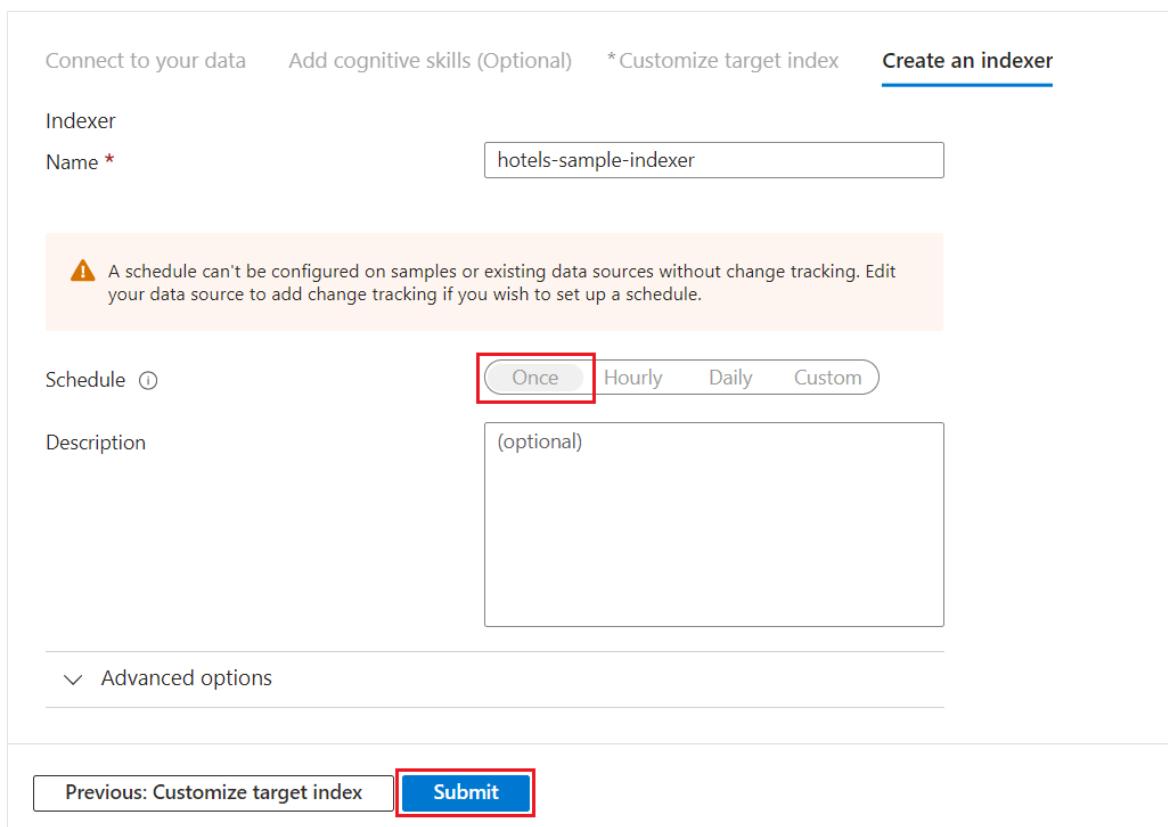
특성은 스토리지에 영향을 줍니다. **Filterable** 필드는 추가 스토리지를 사용하지만 **Retrievable** 필드는 그렇지 않습니다. 자세한 내용은 [특성 및 제안기의 스토리지 영향을 보여 주는 예](#)를 참조하세요.

자동 완성 또는 제안된 쿼리를 원하는 경우 언어 **분석기** 또는 **제안기**를 지정합니다.

## 인덱서 구성 및 실행

마지막 단계에서는 인덱서가 구성되고 실행됩니다. 이 개체는 실행 가능한 프로세스를 정의합니다. 이 단계에서는 데이터 원본, 인덱스 및 인덱서가 만들어집니다.

1. **인덱서 이름**(*hotels-sample-index*)에 대해 시스템에서 생성된 값을 허용합니다.
2. 이 빠른 시작의 경우 기본 옵션을 사용하여 인덱서를 즉시 한 번 실행합니다. 호스트 된 데이터는 정적이므로 변경 내용 추적을 사용할 수 없습니다.
3. **제출**을 선택하여 인덱서를 만드는 동시에 실행합니다.



Connect to your data    Add cognitive skills (Optional)    \*Customize target index    [Create an indexer](#)

Indexer

Name \*

hotels-sample-indexer

**⚠** A schedule can't be configured on samples or existing data sources without change tracking. Edit your data source to add change tracking if you wish to set up a schedule.

Schedule ⓘ

Once    Hourly    Daily    Custom

Description

(optional)

Advanced options

Previous: Customize target index    **Submit**

## 인덱서 진행률 모니터링

포털에서 인덱서 또는 인덱스의 생성을 모니터링할 수 있습니다. 서비스 **개요** 페이지에서는 Azure AI 검색 서비스에서 만든 리소스의 링크를 제공합니다.

1. 왼쪽에서 **인덱서**를 선택합니다.

Status	Name	Last run	Docs succeeded	Errors/Warnings
Success	coffee-indexer	1 year ago	9/9	0/0
In progress	hotels-sample-indexer	2 seconds ago	0/0	0/0
Success	realestate-us-sample-indexer	3 hours ago	4959/4959	0/0

Azure Portal에 페이지 결과가 업데이트되는 데 몇 분 정도 걸릴 수 있습니다. 새로 만든 인덱서가 목록에 진행 중 또는 성공 상태로 표시됩니다. 목록에는 인덱싱된 문서 수도 표시됩니다.

## 검색 인덱스 결과 확인

1. 왼쪽에서 인덱스를 선택합니다.
2. **hotels-sample-index**를 선택합니다.

Azure Portal 페이지가 새로 고쳐질 때까지 기다립니다. 문서 수와 스토리지 크기를 포함하는 인덱스가 표시됩니다.

Name	Document Count	Storage Size
coffee-index	9	99.95 KB
hotels-sample-index	50	335.54 KB
realestate-us-sample-index	4,959	12.21 MB

3. 필드 탭을 선택하여 인덱스 스키마를 볼 수 있습니다.

어떤 쿼리를 작성해야 할지 알 수 있도록 어떤 필드가 **Filterable** 또는 **Sortable**인지 확인합니다.

**hotels-sample-index** ...

X

The screenshot shows the Elasticsearch interface for managing an index named 'hotels-sample-index'. At the top, there are buttons for Save, Discard, Refresh, Create Demo App, Edit JSON, and Delete. Below that, 'Documents' and 'Storage' are listed with values 50 and 335.54 KB respectively. The 'Fields' tab is selected. A search bar at the top of the table says 'Search field names'. The table has columns: Field name, Type, Retrievable, Filterable, Sortable, Facetable, and Searchable. Rows include HotelId (String), HotelName (String), Description (String), Description\_fr (String), Category (String), Tags (StringCollection), ParkingIncluded (Boolean), LastRenovationDate (DateTimeOffset), Rating (Double), Address (ComplexType), and Location (GeographyPoint). Most fields are marked as Retrievable, Filterable, and Facetable.

Field name	Type	Retrievable	Filterable	Sortable	Facetable	Searchable
HotelId	String	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
HotelName	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Description	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Description_fr	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Category	String	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Tags	StringCollection	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ParkingIncluded	Boolean	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
LastRenovationDate	DateTimeOffset	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
Rating	Double	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Address	ComplexType					
Location	GeographyPoint	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

## 필드 추가 또는 변경

필드 탭에서 이름, 지원되는 데이터 형식 및 특성이 포함된 **필드 추가**를 사용하여 새 필드를 만들 수 있습니다.

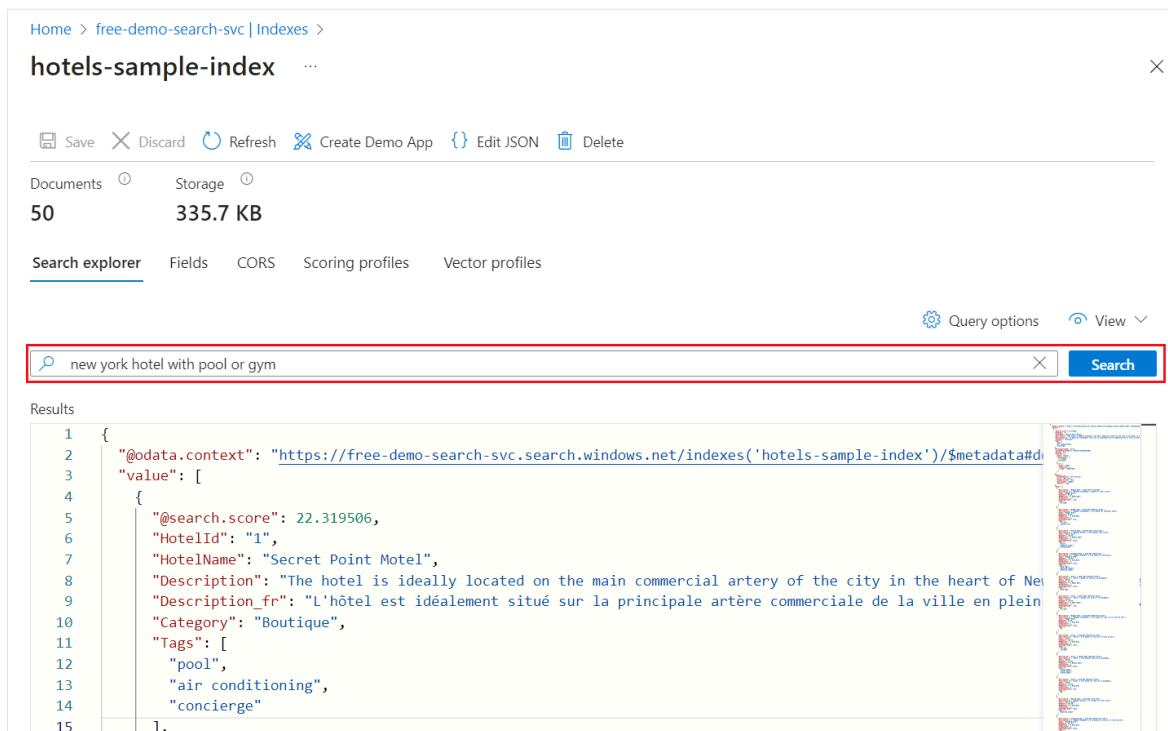
기존 필드를 변경하는 것은 더 어렵습니다. 기존 필드는 인덱스에 물리적 표현이 있으므로 코드에서도 수정할 수 없습니다. 기존 필드를 근본적으로 변경하려면 원본을 대체하는 새 필드를 만들어야 합니다. 점수 매기기 프로필 및 CORS 옵션과 같은 다른 구문을 언제든지 인덱스에 추가할 수 있습니다.

인덱스 디자인 중에 편집할 수 있는 항목과 편집할 수 없는 항목을 명확하게 이해하려면 잠시 시간을 내어 인덱스 정의 옵션을 잠시 살펴봅니다. 필드 목록의 회색 옵션은 수정할 수 없거나 삭제할 수 없는 값을 나타냅니다.

## Search 탐색기를 사용하여 쿼리

이제 검색 탐색기를 사용하여 쿼리할 수 있는 검색 인덱스가 있습니다. 검색 탐색기는 POST 검색 REST API를 준수하는 REST 호출을 보냅니다. 이 도구는 간단한 쿼리 구문과 전체 Lucene 쿼리 구문을 지원합니다.

## 1. 검색 탐색기 탭에서 검색할 텍스트를 입력합니다.



The screenshot shows the Azure Cognitive Search search interface. At the top, there's a breadcrumb navigation: Home > free-demo-search-svc | Indexes >. Below it, the index name "hotels-sample-index" is displayed with a "...". A toolbar at the top has Save, Discard, Refresh, Create Demo App, Edit JSON, and Delete buttons. Under the toolbar, "Documents" (50) and "Storage" (335.7 KB) are shown. Below that, tabs for "Search explorer" (selected), Fields, CORS, Scoring profiles, and Vector profiles are present. On the right, there are "Query options" and "View" dropdowns. A search bar contains the query "new york hotel with pool or gym", which is highlighted with a red box. To the right of the search bar is a "Search" button. The results section below shows a JSON snippet of one result document, with line numbers from 1 to 15 on the left. The result includes fields like @odata.context, search.score, HotelId, HotelName, Description, Description\_fr, Category, and Tags. To the right of the results is a small "Mini-map" view of the index structure, also highlighted with a red box.

## 2. 출력의 보이지 않는 영역으로 빠르게 이동하려면 미니 맵을 사용합니다.



This screenshot shows the same Azure Cognitive Search interface as the previous one, but with a red box highlighting the "Mini-map" feature on the right side of the results pane. The "Results" section on the left displays a JSON snippet of a search result, identical to the one in the previous screenshot. The "Mini-map" is a vertical list of thumbnail previews of all documents in the index, allowing for quick navigation between results.

## 3. 구문을 지정하려면 JSON 보기로 전환합니다.

The screenshot shows the Azure Search Query Explorer. At the top, there are tabs for 'Search explorer', 'Fields', 'CORS', 'Scoring profiles', and 'Vector profiles'. Below the tabs is a search bar containing the query 'new york hotel with pool or gym'. To the right of the search bar are 'Query options' and 'View' dropdown menus. A red box highlights the 'JSON view' button in the 'View' menu. The main area is titled 'Results' and displays a JSON response. The response starts with a line number '1' and a brace '{'. Inside the brace, there is an '@odata.context' field with a URL, a 'value' field which contains an array of objects, and several other fields like '@search.score', 'HotelID', 'HotelName', 'Description', 'Description\_fr', and 'Category'. To the right of the JSON response, there is a small preview pane showing the raw XML or another view of the search results.

## 호텔 샘플 인덱스에 대한 예제 쿼리

다음 예에서는 JSON 뷰와 2024-05-01-preview REST API 버전을 가정합니다.

### 💡 팁

이제 JSON 보기에서는 매개 변수 이름 완료를 위해 Intellisense를 지원합니다. JSON 보기 안에 커서를 놓고 공백 문자를 입력하여 모든 쿼리 매개 변수 목록을 표시하거나 "s"와 같은 단일 문자를 입력하여 "s"로 시작하는 쿼리 매개 변수만 표시합니다. Intellisense는 유효하지 않은 매개 변수를 제외하지 않으므로 최선의 판단을 내립니다.

## 필터 예시

주차, 태그, 리노베이션 날짜, 등급 및 위치를 필터링할 수 있습니다.

JSON

```
{  
  "search": "beach OR spa",  
  "select": "HotelId, HotelName, Description, Rating",  
  "count": true,  
  "top": 10,  
  "filter": "Rating gt 4"  
}
```

부울 필터는 기본적으로 "true"로 가정합니다.

JSON

```
{  
  "search": "beach OR spa",  
  "select": "HotelId, HotelName, Description, Rating",  
  "count": true,
```

```
    "top": 10,  
    "filter": "ParkingIncluded"  
}
```

지리 공간적 검색은 필터 기반입니다. `geo.distance` 함수는 지정된 `Location` 및 `geography'POINT` 좌표를 기반으로 위치 데이터에 대한 모든 결과를 필터링합니다. 이 쿼리는 위도 경도 좌표 `-122.12 47.67` ("Redmond, Washington, USA")에서 5km 이내에 있는 호텔을 찾습니다. 이 쿼리는 호텔 이름 및 주소 위치와 `&count=true` 총 일치 횟수를 표시합니다.

JSON

```
{  
    "search": "*",
    "select": "HotelName, Address/City, Address/StateProvince",
    "count": true,
    "top": 10,
    "filter": "geo.distance(Location, geography'POINT(-122.12 47.67)') le 5"
}
```

## 전체 Lucene 구문 예제

기본 구문은 [간단한 구문](#)이지만 유사 항목 검색이나 용어 승격 또는 정규식을 원하는 경우에는 [전체 구문](#)을 지정합니다.

JSON

```
{  
    "queryType": "full",
    "search": "seattle~",
    "select": "HotelId, HotelName,Address/City, Address/StateProvince",
    "count": true
}
```

기본적으로 `Seattle`에 대한 `seattle`과 같이 철자가 잘못된 쿼리 용어는 일반 검색에서 일치하는 항목을 반환하지 못합니다. `queryType=full` 매개 변수는 물결표 ~ 피연산자를 지원하는 전체 Lucene 쿼리 파서를 호출합니다. 이러한 매개 변수가 있으면 쿼리는 지정된 키워드에 대한 유사 항목 검색을 수행합니다. 쿼리는 키워드와 유사하지만 정확히 일치하지 않는 결과와 함께 일치하는 결과를 찾습니다.

잠시 시간을 내어 인덱스에 대한 이러한 예제 쿼리를 몇 가지 시도해 보세요. 쿼리에 대한 자세한 내용은 [Azure AI 검색에서 쿼리](#)를 참조하세요.

# 리소스 정리

본인 소유의 구독으로 작업하는 경우에는 프로젝트가 끝날 때 여기서 만든 리소스가 계속 필요한지 확인하는 것이 좋습니다. 계속 실행되는 리소스에는 요금이 부과될 수 있습니다. 리소스를 개별적으로 삭제하거나 리소스 그룹을 삭제하여 전체 리소스 세트를 삭제할 수 있습니다.

Azure Portal의 왼쪽 창에 있는 **모든 리소스** 또는 **리소스 그룹**에서 서비스에 대한 리소스를 찾고 관리할 수 있습니다.

무료 서비스를 사용하는 경우 인덱스, 인덱서, 데이터 원본의 세 가지 항목으로 제한됩니다. Azure Portal에서 개별 항목을 삭제하여 제한 이하로 유지할 수 있습니다.

## 다음 단계

브라우저에서 실행되는 즉시 사용할 수 있는 웹앱을 생성하려면 Azure Portal 마법사를 사용해 보세요. 이 빠른 시작에서 만든 작은 인덱스에서 이 마법사를 사용해 보거나 기본 제공 샘플 데이터 세트 중 하나를 사용하여 더 다양한 검색을 경험할 수 있습니다.

[포털에서 데모 앱 만들기](#)

---

## 피드백

이 페이지가 도움이 되었나요?

 Yes

 No

[제품 사용자 의견 제공](#) | [Microsoft Q&A에서 도움말 보기](#)

# 빠른 시작: Azure Portal을 사용하여 텍스트 및 이미지 벡터화

아티클 · 2024. 10. 17.

이 빠른 시작은 Azure Portal에서 [데이터 가져오기 및 벡터화](#) 마법사를 사용하여 [통합 벡터화](#)를 시작하는 데 도움이 됩니다. 이 마법사는 콘텐츠를 청크 분할하고 포함 모델을 호출하여 인덱싱 및 쿼리 중에 콘텐츠를 벡터화합니다.

마법사와 관련된 핵심 사항은 다음과 같습니다.

- 원본 데이터는 Azure Blob Storage, ADLS(Azure Data Lake Storage) Gen2 또는 OneLake 파일 및 바로 가기입니다.
- 문서 구문 분석 모드는 기본값, 즉 Blob 또는 파일당 하나의 검색 문서입니다.
- 인덱스 스키마는 구성할 수 없습니다. 청크 분할된 데이터에 대한 벡터 및 비벡터 필드를 제공합니다.
- 청크는 구성할 수 없습니다. 효과적인 설정은 다음과 같습니다.

JSON

```
textSplitMode: "pages",
maximumPageLength: 2000,
pageOverlapLength: 500
```

## 필수 구성 요소

- Azure 구독 [체험 계정 만들기](#)
- Azure AI와 동일한 지역에 있는 [Azure AI 검색 서비스](#). 기본 계층 이상을 권장합니다.
- [Azure Blob Storage, ADLS\(Azure Data Lake Storage\) Gen2](#)(계층 구조 네임스페이스가 있는 스토리지 계정) 또는 [OneLake lakehouse](#)

Azure Storage는 표준 성능(범용 v2) 계정이어야 합니다. 액세스 계층은 핫, 쿨 및 콜드일 수 있습니다.

- Azure AI Search와 동일한 지역의 Azure AI 플랫폼에 포함된 모델입니다. [배포 지침](#)은 이 문서에 나와 있습니다.

공급자	지원되는 모델
Azure OpenAI Service	text-embedding-ada-002, text-embedding-3-large 또는 text-embedding-3-small.
Azure AI 스튜디오 모델 카탈로그	Azure, Cohere 및 Facebook 포함 모델.
Azure AI 서비스 다중 서비스 계정	이미지 및 텍스트 벡터화를 위한 Azure AI 비전 다중 모델. Azure AI 비전 멀티모달은 선택한 지역에서 사용할 수 있습니다. 업데이트된 목록은 설명서에서 확인하세요. 이 리소스를 사용하려면 계정이 사용 가능한 지역 및 Azure AI 검색과 동일한 지역에 있어야 합니다.

Azure OpenAI 서비스를 사용하는 경우 연결된 [사용자 지정 하위 도메인](#)이 있어야 합니다. Azure Portal을 통해 서비스를 만든 경우 이 하위 도메인은 서비스 설정의 일부로 자동으로 생성됩니다. Azure AI Search 통합과 함께 사용하기 전에 서비스에 사용자 지정 하위 도메인이 포함되어 있는지 확인합니다.

AI Studio에서 만든 Azure OpenAI 서비스 리소스(모델 포함에 대한 액세스 권한 포함)는 지원되지 않습니다. Azure Portal에서 만든 Azure OpenAI 서비스 리소스만 Azure OpenAI 포함 기술 통합과 호환됩니다.

## 공용 엔드포인트 요구 사항

포털 노드가 해당 리소스에 액세스할 수 있도록 이전의 모든 리소스에는 공용 액세스가 사용하도록 설정되어 있어야 합니다. 그렇지 않으면 마법사가 실패합니다. 마법사가 실행된 후 보안을 위해 통합 구성 요소에서 방화벽 및 프라이빗 엔드포인트를 사용하도록 설정할 수 있습니다. 자세한 정보는 [가져오기 마법사의 보안 연결](#)을 참조하세요.

프라이빗 엔드포인트가 이미 있고 이를 사용하지 않도록 설정할 수 없는 경우 대체 옵션은 가상 머신의 스크립트 또는 프로그램에서 해당 엔드포인트 흐름을 실행하는 것입니다. 가상 머신은 프라이빗 엔드포인트와 동일한 가상 네트워크에 있어야 합니다. 통합된 벡터화를 위한 [Python 코드 샘플은 다음과 같습니다](#). 동일한 [GitHub 리포지토리](#)에는 다른 프로그래밍 언어로 된 샘플이 있습니다.

## 역할 기반 액세스 제어 요구 사항

다른 리소스에 검색 서비스를 연결하기 위한 역할 할당을 권장합니다.

1. Azure AI 검색에서 역할을 사용하도록 설정합니다.
2. 관리 ID를 사용하도록 검색 서비스를 구성합니다.

3. 데이터 원본 플랫폼 및 포함 모델 공급자에서 검색 서비스가 데이터 및 모델에 액세스할 수 있도록 하는 역할 할당을 만듭니다. [샘플 데이터 준비](#)에서는 역할을 설정하기 위한 지침을 제공합니다.

무료 검색 서비스는 Azure AI 검색에 대한 연결에서 RBAC를 지원하지만 Azure Storage 또는 Azure AI 비전에 대한 아웃바운드 연결에서는 관리 ID를 지원하지 않습니다. 이 수준의 지원은 무료 검색 서비스와 다른 Azure 서비스 간의 연결에 키 기반 인증을 사용해야 함을 의미합니다.

더 안전한 연결이 필요한 경우:

- 기본 계층 이상을 사용합니다.
- [관리 ID](#)를 구성하고 권한 있는 액세스에 역할을 사용합니다.

#### ① 참고

옵션을 사용할 수 없기 때문에(예: 데이터 원본 또는 포함 모델을 선택할 수 없음) 마법사를 진행할 수 없는 경우 역할 할당을 다시 방문합니다. 오류 메시지는 모델 또는 배포가 존재하지 않음을 나타내지만, 실제로는 검색 서비스에 액세스 권한이 없는 것이 진짜 원인입니다.

## 공간 확인

무료 서비스를 시작하는 경우 인덱스 3개, 데이터 원본 3개, 기술 세트 3개, 인덱서 3개로 제한됩니다. 기본 서비스는 15개로 제한됩니다. 시작하기 전에 추가 항목에 대한 공간이 있는지 확인합니다. 이 빠른 시작에서는 각 개체를 하나씩 만듭니다.

## 의미 순위매기기 확인

마법사는 의미 체계 순위 지정을 지원하지만 기본 계층 이상에서, 그리고 의미 순위매기기가 이미 [검색 서비스에서 활성화](#)된 경우에만 해당합니다. 청구 가능 계층을 사용하는 경우 의미 순위매기기가 사용하도록 설정되어 있는지 확인합니다.

## 샘플 데이터 준비

이 섹션에서는 이 빠른 시작에 적합한 데이터를 알려 줍니다.

Azure Blob Storage

1. Azure 계정으로 [Azure Portal](#)에 로그인하고 Azure Storage 계정으로 이동합니다.
2. 왼쪽 창의 **데이터 스토리지**에서 **컨테이너**를 선택합니다.
3. 새 컨테이너를 만든 다음 이 빠른 시작에 사용된 [상태 계획 PDF 문서](#)를 업로드합니다.
4. 왼쪽 창의 **액세스 제어**에서 [Storage Blob 데이터 읽기 권한자](#) 역할을 검색 서비스 ID에 할당합니다. 또는 [액세스 키](#) 페이지에서 스토리지 계정에 대한 연결 문자열을 가져옵니다.
5. 필요에 따라 컨테이너의 삭제를 검색 인덱스 삭제와 동기화합니다. 다음 단계를 통해 삭제 검색을 위해 인덱서를 구성할 수 있습니다.
  - a. 스토리지 계정에서 [일시 삭제](#)를 사용하도록 설정합니다.
  - b. [네이티브 일시 삭제](#)를 사용하는 경우 Azure Storage에서 추가 단계가 필요하지 않습니다.
  - c. 그렇지 않은 경우 인덱서가 검색할 수 있는 [사용자 지정 메타데이터를 추가](#)하여 삭제용으로 표시된 Blob을 확인합니다. 사용자 지정 속성에 설명이 포함된 이름을 지정합니다. 예를 들어 속성을 이름을 "IsDeleted"로 지정하고 false로 설정할 수 있습니다. 컨테이너의 모든 Blob에 대해 이 작업을 수행합니다. 나중에 Blob을 삭제하려면 속성을 true로 변경합니다. 자세한 내용은 [Azure Storage에서 인덱싱할 때 탐지 변경 및 삭제](#)를 참조하세요.

## 포함 모델 설정

마법사는 Azure OpenAI, Azure AI 비전 또는 Azure AI 스튜디오의 모델 카탈로그에서 배포된 포함 모델을 사용할 수 있습니다.

Azure OpenAI

마법사는 text-embedding-ada-002, text-embedding-3-large, text-embedding-3-small을 지원합니다. 내부적으로 마법사는 [AzureOpenAIEmbedding 기술](#)을 호출하여 Azure OpenAI에 연결합니다.

1. Azure 계정으로 [Azure Portal](#)에 로그인하고 Azure OpenAI 리소스로 이동합니다.

2. 사용 권한 설정:
  - a. 왼쪽 메뉴에서 **액세스 제어**를 선택합니다.
  - b. **추가**를 선택한 다음 **역할 할당 추가**를 선택합니다.
  - c. **작업 함수 역할**에서 **Cognitive Services OpenAI 사용자**,를 선택한 다음, **다음**을 선택합니다.
  - d. **구성원** 아래에서 **관리 ID**를 선택한 다음, **구성원**을 선택합니다.
  - e. 구독 및 리소스 종류(검색 서비스)를 필터링한 다음 검색 서비스의 관리 ID를 선택합니다.
  - f. **검토 + 할당**을 선택합니다.
3. **개요** 페이지에서 엔드포인트 또는 API 키를 복사해야 하는 경우 **엔드포인트를 보려면 여기를 클릭** 또는 **키를 관리하려면 여기를 클릭**을 선택하세요. 키 기반 인증과 함께 Azure OpenAI 리소스를 사용하는 경우 이러한 값을 마법사에 붙여 넣을 수 있습니다.
4. **리소스 관리 및 모델 배포** 아래에서 **배포 관리**를 선택하여 Azure AI Studio를 엽니다.
5. `text-embedding-ada-002`의 배포 이름 또는 지원되는 다른 포함 모델을 복사합니다. 포함 모델이 없는 경우 지금 포함 모델을 배포합니다.

## 마법사 시작

1. Azure 계정으로 [Azure Portal](#)에 로그인하고 Azure AI 검색 서비스로 이동합니다.
2. **개요** 페이지에서 **데이터 가져오기 및 벡터화**를 선택합니다.



## 데이터에 연결

다음 단계는 검색 인덱스에 사용할 데이터 원본에 연결하는 것입니다.

Azure Blob Storage

1. **데이터 연결 설정** 페이지에서 **Azure Blob Storage**를 선택합니다.

2. Azure 구독을 지정합니다.
3. 데이터를 제공하는 스토리지 계정 및 컨테이너를 선택합니다.
4. **삭제 검색** 지원 여부를 지정합니다. 이후 인덱싱 실행 시, Azure Storage에서 일시 삭제된 Blob을 기반으로 검색 문서를 제거하도록 검색 인덱스가 업데이트됩니다.
  - Blob은 **네이티브 Blob 일시 삭제** 또는 **사용자 지정 데이터**를 사용하여 일시 삭제를 지원합니다.
  - 이전에 Azure Storage에서 **일시 삭제를 사용하도록 설정**했어야 하며, 선택적으로 인덱싱이 삭제 플래그로 인식할 수 있는 **사용자 지정 메타데이터**를 **추가**했어야 합니다. 이러한 단계에 대한 자세한 내용은 **샘플 데이터 준비**를 참조하세요.
  - Blob을 **사용자 지정 데이터**를 사용하여 일시 삭제용으로 구성한 경우 이 단계에서 메타데이터 속성 이름 값 쌍을 제공합니다. "IsDeleted"를 사용하는 것이 좋습니다. Blob에서 "IsDeleted"가 true로 설정된 경우 인덱서는 다음 인덱서 실행 시 해당 검색 문서를 삭제합니다.

마법사는 Azure Storage에서 유효한 설정을 확인하지 않거나 요구 사항이 충족되지 않으면 오류를 throw합니다. 대신 삭제 탐지가 작동하지 않으며 검색 인덱스는 시간이 지남에 따라 분리된 문서를 수집할 가능성이 높습니다.

**Configure your Azure Blob Storage**

Connect to your Azure Blob Storage containing PDFs and other unstructured data files. [Learn more](#)

Subscription *	My Azure Subscription
Storage account *	My Azure Storage Account
Blob container * ⓘ	health-plan-pdfs
Blob folder ⓘ	your/folder/here
<input checked="" type="checkbox"/> Enable deletion tracking ⓘ	
<input type="radio"/> Native blob soft delete	
<input checked="" type="radio"/> Soft delete using custom metadata	
Soft delete column *	IsDeleted
Delete marker value *	true
<input checked="" type="checkbox"/> Authenticate using managed identity. <a href="#">Learn more</a>	
Managed identity type ⓘ	System-assigned

5. 검색 서비스가 **관리 ID**를 사용하여 Azure Storage에 **연결**할지 여부를 지정합니다.

- 시스템 관리 ID 또는 사용자 관리 ID를 선택하라는 메시지가 표시됩니다.
- ID에는 Azure Storage에 대한 **Storage Blob 데이터 읽기 권한자** 역할이 있어야 합니다.
- 이 단계를 건너뛰지 마세요. 마법사가 Azure Storage에 연결할 수 없는 경우 인덱싱하는 동안 연결 오류가 발생합니다.

6. 다음을 선택합니다.

## 텍스트 벡터화

이 단계에서는 청크 분할 데이터를 벡터화하기 위한 포함 모델을 지정합니다.

1. 텍스트 벡터화 페이지에서 포함 모델의 원본을 선택합니다.

- Azure OpenAI
- Azure AI 스튜디오 모델 카탈로그
- Azure AI 검색과 동일한 지역에 있는 기존 Azure AI 비전 다중 모달 리소스입니다. 동일한 지역에 [Azure AI 서비스 다중 서비스 계정](#)이 없는 경우 이 옵션을 사용할 수 없습니다.

2. Azure 구독을 선택합니다.

3. 리소스에 따라 선택합니다.

- Azure OpenAI의 경우 text-embedding-ada-002, text-embedding-3-large 또는 text-embedding-3-small의 기존 배포를 선택합니다.
- AI 스튜디오 카탈로그의 경우 Azure, Cohere 및 Facebook 포함 모델의 기존 배포를 선택합니다.
- AI 비전 멀티모달 포함 기술의 경우 계정을 선택합니다.

자세한 내용은 이 문서 앞부분의 [포함 모델 설정](#)을 참조하세요.

4. 검색 서비스가 API 키 또는 관리 ID를 사용하여 인증할지 여부를 지정합니다.

- ID에는 Azure AI 다중 서비스 계정에 대한 **Cognitive Services OpenAI 사용자** 역할이 있어야 합니다.

5. 이러한 리소스 사용이 청구에 미치는 영향을 인정하는 확인란을 선택합니다.

6. 다음을 선택합니다.

# 이미지 벡터화 및 보강

콘텐츠에 이미지가 포함된 경우 다음 두 가지 방법으로 AI를 적용할 수 있습니다.

- 카탈로그에서 지원되는 이미지 포함 모델을 사용하거나 Azure AI Vision 멀티모달 포함 API를 선택하여 이미지를 벡터화합니다.
- OCR(광학 인식)을 사용하여 이미지의 텍스트를 인식합니다. 이 옵션은 이미지에서 텍스트를 읽기 위해 [OCR 기술](#)을 호출합니다.

Azure AI 검색 및 Azure AI 리소스는 동일한 지역에 있어야 합니다.

1. **이미지 벡터화** 페이지에서 마법사가 수행해야 하는 연결 종류를 지정합니다. 이미지 벡터화를 위해 마법사는 Azure AI 스튜디오 또는 Azure AI 비전의 포함 모델에 연결할 수 있습니다.
2. 구독을 지정합니다.
3. Azure AI Studio 모델 카탈로그의 경우 프로젝트와 배포를 지정합니다. 자세한 내용은 이 문서 앞부분의 [포함 모델 설정](#)을 참조하세요.
4. 선택적으로 이진 파일 이미지(예: 검사한 문서 파일)를 해독하고 [OCR을 사용](#)하여 텍스트를 인식할 수 있습니다.
5. 이러한 리소스 사용이 청구에 미치는 영향을 인정하는 확인란을 선택합니다.
6. **다음을 선택합니다.**

## 고급 설정 선택

1. 고급 설정 페이지에서 선택적으로 [의미 체계 순위 지정](#)을 추가하여 쿼리 실행이 끝날 때 결과의 순위 지정을 다시 지정할 수 있습니다. 순위 지정을 다시 매기면 의미체계상 가장 관련성이 높은 일치 항목이 상위로 승격됩니다.
2. 선택적으로 인덱서의 [실행 일정](#)을 지정합니다.
3. **다음을 선택합니다.**

## 마법사 마침

1. **구성 검토** 페이지에서 마법사가 만들 개체의 접두사를 지정합니다. 공통 접두사는 정리하는 데 도움이 됩니다.
2. **만들기를 실행합니다.**

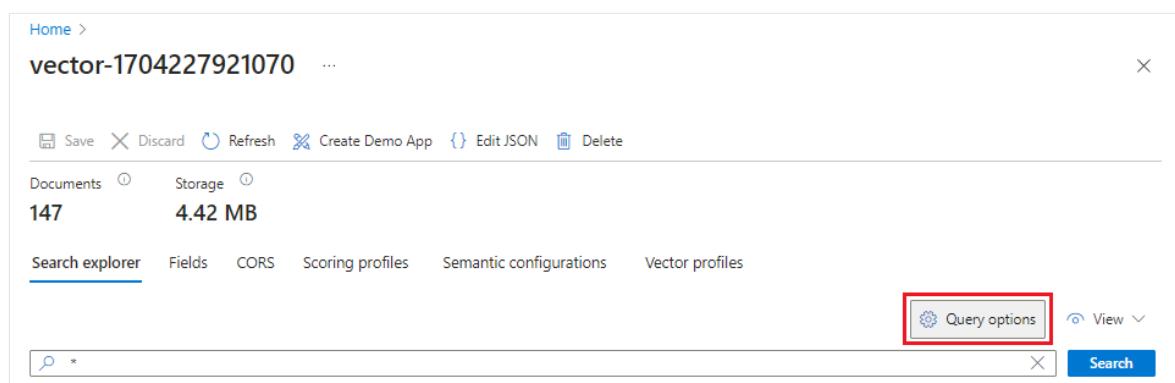
마법사가 구성을 완료하면 다음 개체가 만들어집니다.

- 데이터 원본을 연결합니다.
- 벡터 필드, 벡터라이저, 벡터 프로필 및 벡터 알고리즘으로 인덱스를 생성합니다. 마법사 워크플로 중에는 기본 인덱스를 설계하거나 수정할 수 없습니다. 인덱스는 2024-05-01-preview REST API를 준수합니다.
- 청킹을 위한 텍스트 분할 기술과 벡터화를 위한 포함 기술을 갖춘 기술 세트입니다. 포함 기술은 Azure OpenAI용 AzureOpenAIEmbeddingModel 기술 또는 Azure AI 스튜디오 모델 카탈로그용 AML 기술입니다. 또한 기술 세트에는 데이터 원본의 한 문서에서 "자식" 인덱스의 해당 청크로 데이터를 매핑할 수 있는 인덱스 프로젝션 구성도 있습니다.
- 필드 매핑 및 출력 필드 매핑(해당되는 경우)이 있는 인덱서입니다.

## 결과 확인

검색 탐색기는 텍스트 문자열을 입력으로 받아들인 다음 벡터 쿼리 실행을 위해 텍스트를 벡터화합니다.

- Azure Portal에서 검색 관리>인덱스로 이동한 다음 만든 인덱스를 선택합니다.
- 필요에 따라 쿼리 옵션을 선택하고 검색 결과에서 벡터 값을 숨깁니다. 이 단계를 수행하면 검색 결과를 더 쉽게 읽을 수 있습니다.



The screenshot shows the Azure Portal interface for managing a vector index named 'vector-1704227921070'. The top navigation bar includes 'Home', 'Save', 'Discard', 'Refresh', 'Create Demo App', 'Edit JSON', and 'Delete'. Below the navigation is a summary section showing 'Documents: 147' and 'Storage: 4.42 MB'. The main content area has tabs for 'Search explorer', 'Fields', 'CORS', 'Scoring profiles', 'Semantic configurations', and 'Vector profiles'. A 'Query options' button is highlighted with a red box. At the bottom is a search bar with a 'Search' button.

- 보기 메뉴에서 JSON 보기 를 선택하면 `text` 벡터 쿼리 매개 변수에 벡터 쿼리에 대한 텍스트를 입력할 수 있습니다.

Dashboard > free-demo-search-svc > Import and vectorize data >

# vector-1701309912002

...

Save Discard Refresh Create Demo App Edit JSON Delete

Documents Storage ...

0 **0 Bytes**

Search explorer Fields CORS Scoring profiles Vector profiles

2023-10-01-Preview View

JSON query editor

```

1  {
2    "search": "*",
3    "select": "chunk, title",
4    "vectorQueries": [
5      {
6        "kind": "text",
7        "text": "Which plan has the lowest deductible?", highlighted
8        "k": 5,
9        "fields": "vector"
10   }

```

Query view highlighted

JSON view highlighted

Search

Results

```

1  {
2    "@odata.context": "https://free-demo-search-svc.search.windows.net/indexes('vector-1701309912002')/$metadata#",
3    "value": [
4      {
5        "@search.score": 0.835181,
6        "chunk": "year deductible is the same for \n\nall members of the plan and is reset each year on the plan",
7        "title": "Northwind_Health_Plus_Benefits_Details.pdf"
8      },

```

마법사는 `vector` 필드에 대해 벡터 쿼리를 실행하고 5개의 가장 인접한 항목을 반환하는 기본 쿼리를 제공합니다. 벡터 값을 숨기도록 선택한 경우 기본 쿼리에는 검색 결과에서 `vector` 필드를 제외하는 `select` 문이 포함됩니다.

JSON

```

{
  "select": "chunk_id,parent_id,chunk,title",
  "vectorQueries": [
    {
      "kind": "text",
      "text": "*",
      "k": 5,
      "fields": "vector"
    }
  ]
}

```

4. `text` 값의 경우 별표(\*)를 상태 계획과 관련된 질문(예: `Which plan has the lowest deductible?`)으로 바꿉니다.
5. 쿼리를 실행하려면 **쿼리**를 선택합니다.

Search explorer Fields CORS Scoring profiles Semantic configurations Vector profiles

Query options View

which plan has the lowest deductible

Results

```
3 "value": [
4   {
5     "@search.score": 0.8322106,
6     "chunk_id": "55ffd790d9b2_aHR0cHM6Ly9oZwlkaXN0ZlWvc3RvcmFnZS5ibG9iLmNvcnUud2luZG93cy5uZXQvaGVhbHRoLX
7     "parent_id": "aHR0cHM6Ly9oZwlkaXN0ZlWvc3RvcmFnZS5ibG9iLmNvcnUud2luZG93cy5uZXQvaGVhbHRoLXBsYn4tcGRmcy
8     "chunk": "year deductible is the same for \nall members of the plan and is reset each year on the
9     "title": "Northwind_Health_Plus_Benefits_Details.pdf"
10   },
11   {
12     "@search.score": 0.8314516,
13     "chunk_id": "2d96eb5d5836_aHR0cHM6Ly9oZwlkaXN0ZlWvc3RvcmFnZS5ibG9iLmNvcnUud2luZG93cy5uZXQvaGVhbHRoLX
14     "parent_id": "aHR0cHM6Ly9oZwlkaXN0ZlWvc3RvcmFnZS5ibG9iLmNvcnUud2luZG93cy5uZXQvaGVhbHRoLXBsYn4tcGRmcy
15     "chunk": "to all services. For example, you may not \n\ne subject to the deductible when you receiv
16     "title": "Northwind_Standard_Benefits_Details.pdf"
17   },
18   {
19     "@search.score": 0.82255286,
20     "chunk_id": "2d96eb5d5836_aHR0cHM6Ly9oZwlkaXN0ZlWvc3RvcmFnZS5ibG9iLmNvcnUud2luZG93cy5uZXQvaGVhbHRoLX
21     "parent_id": "aHR0cHM6Ly9oZwlkaXN0ZlWvc3RvcmFnZS5ibG9iLmNvcnUud2luZG93cy5uZXQvaGVhbHRoLXBsYn4tcGRmcy
22     "chunk": "should compare the \n\nout-of-pocket maximums and deductibles of different plans before de
23     "title": "Northwind_Standard_Benefits_Details.pdf"
24   },
25   {
26     "@search.score": 0.8215061,
27     "chunk_id": "55ffd790d9b2_aHR0cHM6Ly9oZwlkaXN0ZlWvc3RvcmFnZS5ibG9iLmNvcnUud2luZG93cy5uZXQvaGVhbHRoLX
28     "parent_id": "aHR0cHM6Ly9oZwlkaXN0ZlWvc3RvcmFnZS5ibG9iLmNvcnUud2luZG93cy5uZXQvaGVhbHRoLXBsYn4tcGRmcy
```

5개의 일치 항목이 나타나야 합니다. 각 문서는 원본 PDF의 일부입니다. `title` 필드는 청크가 어느 PDF에서 나오는지 보여 줍니다.

6. 특정 문서의 모든 청크를 보려면 특정 PDF의 title 필드에 필터를 추가합니다.

```
JSON

{
  "select": "chunk_id,parent_id,chunk,title",
  "filter": "title eq 'Benefit_Options.pdf'",
  "count": true,
  "vectorQueries": [
    {
      "kind": "text",
      "text": "*",
      "k": 5,
      "fields": "vector"
    }
  ]
}
```

## 정리

Azure AI 검색은 청구 가능한 리소스입니다. 더 이상 필요하지 않은 경우 요금이 부과되지 않도록 구독에서 삭제합니다.

# 다음 단계

이 빠른 시작에서는 통합 벡터화에 필요한 모든 개체를 만드는 [데이터 가져오기 및 벡터화](#) 마법사를 소개했습니다. 각 단계를 자세히 살펴보려면 [통합 벡터화 샘플](#)을 사용해 보세요.

---

## 피드백

이 페이지가 도움이 되었나요?

 Yes

 No

[제품 사용자 의견 제공](#) | Microsoft Q&A에서 도움말 보기

# 빠른 시작: Azure Portal에서 검색 탐색기를 사용하여 이미지 검색

아티클 • 2024. 10. 20.

이 빠른 시작에서는 Azure Portal에서 **데이터 가져오기 및 벡터화** 마법사를 사용하여 이미지 검색을 시작하는 방법을 보여 줍니다. 또한 검색 탐색기를 사용하여 이미지 기반 쿼리를 실행하는 방법을 보여 줍니다.

샘플 데이터는 [azure-search-sample-data](#) 리포지토리의 이미지 파일로 구성되지만 다른 이미지를 사용하여 연습을 계속 진행할 수도 있습니다.

## 필수 구성 요소

- Azure 구독 [체험 계정 만들기](#)
- 이미지 벡터화 및 OCR(광학 인식)에 사용되는 [Azure AI 서비스 다중 서비스 계정](#)입니다. 이미지 벡터화에는 Azure AI Vision 멀티모달 포함이 필요합니다. 업데이트된 [지역 목록은 설명서를 확인하세요](#).
- 인덱싱 및 쿼리를 위한 Azure AI 검색. 모든 계층에 있을 수 있지만 Azure AI 다중 서비스와 [동일한 지역에 있어야 합니다](#).

서비스 계층에 따라 인덱싱할 수 있는 Blob 수가 결정됩니다. 여기에서는 이 연습을 만드는데 무료 계층을 사용하였으며 콘텐츠를 10개의 JPG 파일로 제한했습니다.

- 이미지 파일을 Blob으로 저장하기 위한 Azure Storage. 계층 구조 네임스페이스가 있는 스토리지 계정인 Azure Blob Storage 또는 Azure Data Lake Storage Gen2, 즉 표준 성능(범용 v2) 계정을 사용합니다. 액세스 계층은 핫, 쿨 및 콜드일 수 있습니다.

포털 노드가 해당 리소스에 액세스할 수 있도록 이전의 모든 리소스에는 공용 액세스가 사용하도록 설정되어 있어야 합니다. 그렇지 않으면 마법사가 실패합니다. 마법사가 실행된 후 보안을 위해 통합 구성 요소에서 방화벽 및 프라이빗 엔드포인트를 사용하도록 설정할 수 있습니다. 자세한 정보는 [가져오기 마법사의 보안 연결](#)을 참조하세요.

프라이빗 엔드포인트가 이미 있고 이를 사용하지 않도록 설정할 수 없는 경우 대체 옵션은 가상 머신의 스크립트 또는 프로그램에서 해당 엔드투엔드 흐름을 실행하는 것입니다. 가상 머신은 프라이빗 엔드포인트와 동일한 가상 네트워크에 있어야 합니다. 통합된 벡터화를 위한 [Python 코드 샘플은 다음과 같습니다](#). 동일한 [GitHub 리포지토리](#)에는 다른 프로그래밍 언어로 된 샘플이 있습니다.

무료 검색 서비스는 Azure AI 검색에 대한 연결에서 역할 기반 액세스 제어를 지원하지만 Azure Storage 또는 Azure AI 비전에 대한 아웃바운드 연결에서 관리 ID를 지원하지 않습니다. 이 수준의 지원은 무료 검색 서비스와 다른 Azure 서비스 간의 연결에 키 기반 인증을 사용해야 함을 의미합니다. 보다 안전한 연결의 경우:

- 기본 계층 이상을 사용합니다.
- 다른 Azure 서비스에서 Azure AI 검색의 요청을 허용하도록 [관리 ID를 구성](#)하고 역할 할당을 구성합니다.

## 공간 확인

무료 서비스를 시작하는 경우 인덱스 3개, 데이터 원본 3개, 기술 세트 3개, 인덱서 3개로 제한됩니다. 시작하기 전에 추가 항목에 대한 공간이 있는지 확인합니다. 이 빠른 시작에서는 각 개체를 하나씩 만듭니다.

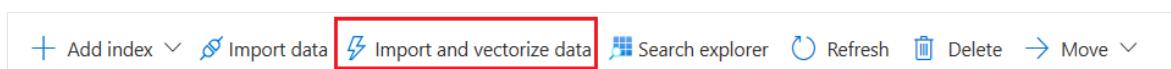
## 샘플 데이터 준비

1. [unsplash-signs 이미지 폴더](#)를 로컬 폴더에 다운로드하거나 자체 이미지를 찾습니다. 무료 검색 서비스에서 보강 처리를 위한 무료 할당량을 초과하지 않도록 이미지 파일을 20개 미만으로 유지합니다.
2. Azure 계정으로 [Azure Portal](#)에 로그인하고 Azure Storage 계정으로 이동합니다.
3. 왼쪽 창의 **데이터 스토리지**에서 컨테이너를 선택합니다.
4. 새 컨테이너를 만든 다음 이미지를 업로드합니다.

## 마법사 시작

검색 서비스와 Azure AI 서비스가 동일한 [지원 지역](#) 및 테넌트에 있고 Azure Storage Blob 컨테이너가 기본 구성을 사용하고 있다면 마법사를 시작할 준비가 된 것입니다.

1. Azure 계정으로 [Azure Portal](#)에 로그인하고 Azure AI 검색 서비스로 이동합니다.
2. **개요** 페이지에서 **데이터 가져오기 및 벡터화**를 선택합니다.



## 데이터에 연결

다음 단계는 이미지를 제공하는 데이터 원본에 연결하는 것입니다.

1. 데이터 연결 설정 페이지에서 Azure Blob Storage를 선택합니다.
2. Azure 구독을 지정합니다.
3. Azure Storage의 경우 데이터를 제공하는 계정 및 컨테이너를 선택합니다. 나머지 상자에는 기본값을 사용합니다.

**Set up your data connection**

Connect to an external Azure Blob Storage or OneLake containing PDFs and other unstructured data files. [Learn more](#)

 **Azure Blob Storage**  
Azure

 **OneLake files**  
Fabric

Subscription *	<input type="text" value="My Azure subscription"/>
Blob storage account *	<input type="text" value="mydemostorageeastus"/>
Blob container * ⓘ	<input type="text" value="unsplash-images"/>
Blob folder ⓘ	<input type="text" value="your/folder/here"/>
<input type="checkbox"/> Enable deletion tracking ⓘ	
<input type="checkbox"/> Authenticate using managed identity. <a href="#">Learn more</a>	

4. 다음을 선택합니다.

## 텍스트 벡터화

원시 콘텐츠에 텍스트가 포함되어 있거나 기술 세트가 텍스트를 생성하는 경우 마법사는 텍스트 포함 모델을 호출하여 해당 콘텐츠에 대한 벡터를 생성합니다. 이 연습에서는 다음 단계에서 추가하는 OCR 기술에서 텍스트가 생성됩니다.

Azure AI 비전은 텍스트 포함을 제공하므로 텍스트 벡터화에 해당 리소스를 사용합니다.

1. 텍스트 벡터화 페이지에서 AI 비전 벡터화를 선택합니다. 사용할 수 없는 경우 Azure AI 검색과 Azure AI 다중 서비스 계정이 [AI 비전 다중 모달 API를 지원](#)하는 지역에 함께 있는지 확인합니다.

**Vectorize your text**

Connect to an Azure OpenAI, AI Studio or an Azure AI service and select an embedding model or multi-service account for vector generation. [Learn more](#)

Kind	<input type="text" value="AI Vision vectorization"/>
Subscription *	<input type="text" value="My Azure subscription"/>
Select a multi-service account *	<input type="text" value="my-demo-azure-ai-multiservice"/> 
<input checked="" type="checkbox"/> I acknowledge that including Azure AI services involves a Pay-as-you-go pricing model. <a href="#">View pricing</a>	

2. 다음을 선택합니다.

## 이미지 벡터화 및 보강

Azure AI 비전을 사용하여 이미지 파일의 벡터 표현을 생성합니다.

이 단계에서는 AI를 적용하여 이미지에서 텍스트를 추출할 수도 있습니다. 마법사는 Azure AI 서비스의 OCR을 사용하여 이미지 파일의 텍스트를 인식합니다.

OCR이 워크플로에 추가되면 인덱스에 두 개의 출력이 더 표시됩니다.

- `chunk` 필드는 이미지에 있는 텍스트의 OCR 생성 문자열로 채워집니다.
- `text_vector` 필드는 `chunk` 문자열을 나타내는 포함으로 채워집니다.

`chunk` 필드에 일반 텍스트를 포함시키는 것은 [의미 체계 순위 지정 및 점수 매기기](#) 프로필과 같이 문자열에 대해 작동하는 관련성 기능을 사용하려는 경우 유용합니다.

1. 이미지 벡터화 페이지에서 **이미지 벡터화** 확인란을 선택한 다음, **AI 비전 벡터화**를 선택합니다.
2. 텍스트 벡터화에서 선택한 것과 동일한 AI 서비스 사용을 선택합니다.
3. 보강 섹션에서 **이미지에서 텍스트 추출** 및 **이미지 벡터화**에 선택한 것과 동일한 AI 서비스 사용을 선택합니다.

The screenshot shows the 'Vectorize your images' configuration section. It includes a description, a checked checkbox for 'Vectorize images', a dropdown menu set to 'AI Vision vectorization', and another checked checkbox for 'Use same AI service selected for text vectorization'. Below this is the 'Enrich your data with AI skills' section, which includes a description, a checked checkbox for 'Extract text from images', and another checked checkbox for 'Use same AI service selected for image vectorization'.

4. 다음을 선택합니다.

## 새 필드 맵핑

고급 설정 페이지에서 필요에 따라 새 필드를 추가할 수 있습니다. 기본적으로 마법사는 이러한 특성을 사용하여 다음 필드를 생성합니다.

#### 테이블 확장

필드	적용 대상	설명
chunk_id	텍스트 및 이미지 벡터	생성된 문자열 필드입니다. 검색 가능, 검색 가능, 정렬 가능. 인덱스 문서 키입니다.
text_parent_id	이미지 벡터	생성된 문자열 필드입니다. 검색 가능하고 필터링할 수 있습니다. 청크가 시작되는 부모 문서를 식별합니다.
image_parent_id	이미지 벡터	생성된 문자열 필드입니다. 검색 가능하고 필터링할 수 있습니다. 이미지가 시작되는 부모 문서를 식별합니다.
chunk	텍스트 및 이미지 벡터	문자열 필드입니다. 사람이 읽을 수 있는 데이터 청크 버전입니다. 검색 가능하고 검색 가능하지만 필터링할 수 없거나 패싯 가능하거나 정렬할 수 없습니다.
title	텍스트 및 이미지 벡터	문자열 필드입니다. 사람이 읽을 수 있는 문서 제목 또는 페이지 제목 또는 페이지 번호입니다. 검색 가능하고 검색 가능하지만 필터링할 수 없거나 패싯 가능하거나 정렬할 수 없습니다.
image_vector	이미지 벡터	Collection(Edm.single). 이미지의 벡터 표현입니다. 검색 가능하고 검색 가능하지만 필터링할 수 없거나 패싯 가능하거나 정렬할 수 없습니다.

생성된 필드 또는 해당 특성을 수정할 수는 없지만 데이터 원본에서 제공하는 경우 새 필드를 추가할 수 있습니다. 예를 들어 Azure Blob Storage는 메타데이터 필드 컬렉션을 제공합니다.

1. 새로 추가를 선택합니다.
2. 사용 가능한 필드 목록에서 원본 필드를 선택하고, 인덱스의 필드 이름을 제공하고, 필요에 따라 기본 데이터 형식을 적용하거나 재정의합니다.  
메타데이터 필드는 검색할 수 있지만 검색할 수 없거나, 필터링 가능하거나, 패싯 가능하거나, 정렬할 수 없습니다.
3. 스키마를 원래 버전으로 복원하려면 다시 설정을 선택합니다.

## 인덱싱 예약

- 고급 설정 페이지의 인덱싱 일정에서 인덱서에 대한 실행 일정을 지정합니다. 이 연습에서는 한 번을 권장합니다. 기본 데이터가 일시적인 데이터 원본의 경우 인덱싱을 예약하여 변경 내용을 적용할 수 있습니다.



- 다음을 선택합니다.

## 마법사 마침

- 구성 검토 페이지에서 마법사가 만들 개체의 접두사를 지정합니다. 공통 접두사는 정리하는 데 도움이 됩니다.

Objects name prefix  
vector-image-search

Review your configuration

Vectorize your text

Attached AI Services - Vision resource	my-demo-azure-ai-multiservice
Vectorize your images	
Attached AI Services - Vision resource	my-demo-azure-ai-multiservice
Others	
Extracting text from images	Enabled
Semantic ranker	Enabled
Indexer run schedule	Once

- 만들기를 실행합니다.

마법사가 구성을 완료하면 다음 개체가 만들어집니다.

- 인덱싱 파이프라인을 구동하는 인덱서
- Azure Blob Storage에 대한 데이터 원본 연결입니다.
- 벡터 필드, 텍스트 필드, 벡터라이저, 벡터 프로필 및 벡터 알고리즘이 포함된 인덱스. 마법사 워크플로 중에는 기본 인덱스를 수정할 수 없습니다. 인덱스는 [2024-05-01-preview REST API](#)를 준수하므로 미리 보기 기능을 사용할 수 있습니다.

- 다음 5가지 기술을 갖춘 기술 세트:
  - [OCR](#) 기술은 이미지 파일에서 텍스트를 인식합니다.
  - [텍스트 병합](#) 기술은 OCR 처리의 다양한 출력을 통합합니다.
  - [텍스트 분할](#) 기술은 데이터 청크화를 추가합니다. 이 기술은 마법사 워크플로에서 기본 제공됩니다.
  - [Azure AI 비전 다중 모드 포함](#) 기술은 OCR에서 생성된 텍스트를 벡터화하는데 사용됩니다.
  - 이미지를 벡터화하기 위해 [Azure AI 비전 다중 모드 포함](#) 기술이 다시 호출됩니다.

## 결과 확인

검색 탐색기는 쿼리 입력으로 텍스트, 벡터, 이미지를 허용합니다. 이미지를 검색 영역으로 끌거나 선택할 수 있습니다. 검색 탐색기는 이미지를 벡터화하고 검색 엔진에 쿼리 입력으로 벡터를 보냅니다. 이미지 벡터화에서는 인덱스에 [데이터 가져오기 및 벡터화](#) 기능이 포함 모델 입력을 기반으로 생성하는 벡터라이저 정의가 있다고 가정합니다.

1. Azure Portal에서 [검색 관리](#)>[인덱스](#)로 이동한 다음 만든 인덱스를 선택합니다. 검색 탐색기는 첫 번째 탭입니다.
2. 보기 메뉴에서 [이미지 보기](#)를 선택합니다.

The screenshot shows the Azure portal interface for managing search indexes. The top navigation bar includes 'Home', 'free-search-demo-svc | Indexes', and a search bar. Below the navigation is a toolbar with 'Save', 'Discard', 'Refresh', 'Create demo app', 'Edit JSON', and 'Delete' buttons. The main content area displays index statistics: 'Documents' (20), 'Total storage' (351.73 KB), 'Vector index size' (244.1 KB), and 'Max storage' (50 MB). Below these stats are tabs for 'Search explorer', 'Fields', 'CORS', 'Scoring profiles', and 'Vector profiles'. A 'View' dropdown menu at the top right contains three options: 'Query view' (selected), 'Image view' (highlighted with a red box), and 'JSON view'. At the bottom left, there's a 'Results' section showing a count of 1.

3. 샘플 이미지 파일이 포함된 로컬 폴더에서 이미지를 끌어옵니다. 아니면 파일 브라우저를 열어 로컬 이미지 파일을 선택합니다.
4. 쿼리를 실행하려면 [쿼리를 선택합니다](#).

검색한 이미지가 가장 일치해야 합니다. [벡터 검색](#)은 유사한 벡터와 일치하므로 검색 엔진은 쿼리 입력과 충분히 유사한 모든 문서(최대  $k$  개 결과)를 반환합니다. JSON 보기로 전환하여 관련성 조정이 포함된 고급 쿼리를 볼 수 있습니다.

Search explorer Fields CORS Scoring profiles Vector profiles

Query options View ▾

Drag and drop an image or [Browse image file](#)

**Search**

Results

```

1  {
2    "@odata.context": "https://free-search-demo-svc.search.windows.net/indexes('vector-image-search')/$metadata#d
3    "value": [
4      {
5        "@search.score": 0.636817,
6        "chunk_id": "222ab60cd608_aHR0cHM6Ly9teWR1bw9zdG9yYWd1ZWfdHVzLmJsb2IuY29yZS53aW5kb3dzLm5ldC91bnNwbGFzaC1j
7        "text_parent_id": "aHR0cHM6Ly9teWR1bw9zdG9yYWd1ZWfdHVzLmJsb2IuY29yZS53aW5kb3dzLm5ldC91bnNwbGFzaC1pbWFnZXI
8        "chunk": "TACOS VS BURRITOS",
9        "title": "matt-nelson-CTvIvzQs1E-unsplash.jpg",
10       "metadata_storage_path": "https://mydemostorageeastus.blob.core.windows.net/unsplash-images/matt-nelson-c
11       "image_parent_id": null,
12       "text_vector": [
13         -0.0031604767,
14         0.0025501251,

```

## 5. 다른 쿼리 옵션을 사용하여 검색 결과를 비교해 보세요.

- 결과의 가독성을 높이기 위해 벡터를 숨깁니다(권장).
- 쿼리할 벡터 필드를 선택합니다. 기본값은 텍스트 벡터이지만 이미지 벡터를 지정하여 쿼리 실행에서 텍스트 벡터를 제외하도록 할 수 있습니다.

## 정리

이 데모에서는 청구 가능한 Azure 리소스를 사용합니다. 리소스가 더 이상 필요하지 않은 경우 요금이 부과되지 않도록 구독에서 삭제합니다.

## 다음 단계

이 빠른 시작에서는 이미지 검색에 필요한 모든 개체를 만드는 [데이터 가져오기 및 벡터화](#) 마법사를 소개했습니다. 각 단계를 자세히 살펴보려면 [통합 벡터화 샘플](#) 을 사용해 보세요.

## 피드백

이 페이지가 도움이 되었나요?

Yes

No

[제품 사용자 의견 제공](#) | [Microsoft Q&A에서 도움말 보기](#)

# 빠른 시작: Azure Portal에서 데모 앱 만들기

아티클 • 2024. 09. 03.

이 Azure AI 검색 빠른 시작에서는 Azure Portal의 [데모 앱 만들기](#) 마법사를 사용하여 브라우저에서 실행되는 다운로드 가능한 "localhost" 스타일 웹앱을 생성합니다. 구성에 따라 생성된 앱은 처음 사용할 때 검색 서비스의 인덱스에 대한 라이브 읽기 전용 연결을 사용하여 작동합니다. 기본 앱에는 검색 창, 결과 영역, 사이드바 필터 및 자동 완성 지원이 포함될 수 있습니다.

데모 앱은 클라이언트 앱에서 인덱스가 작동하는 방식을 시각화하는데 도움을 줄 수 있지만, 프로덕션 시나리오에는 적합하지 않습니다. 프로덕션 앱에는 데모 앱에서 제공하지 않는 보안, 오류 처리, 호스팅 논리가 포함되어야 합니다.

## 필수 조건

시작하기 전에 다음과 같은 필수 구성 요소를 갖추어야 합니다.

- 활성 구독이 있는 Azure 계정. [체험 계정을 만듭니다](#).
- Azure AI 검색 서비스. [서비스를 만들거나](#) 현재 구독에서 [기존 서비스를 찾습니다](#). 이 빠른 시작에서는 체험 서비스를 사용할 수 있습니다.
- Microsoft Edge([최신 버전](#)) 또는 Google Chrome.
- 생성된 애플리케이션의 기반으로 사용할 [검색 인덱스](#)입니다.

이 빠른 시작에서는 미리 보기 이미지가 있으므로 기본 제공 부동산 샘플 데이터와 인덱스를 사용합니다(마법사에서 결과 페이지에 이미지 추가를 지원함). 이 연습에 사용되는 인덱스를 만들려면 [데이터 가져오기](#) 마법사를 실행하여 `realestate-us-sample` 데이터 원본을 선택합니다.

**Import data**

Connect to your data Add cognitive skills (Optional) Customize target index ...

Create and load a search index using data from an existing Azure data source in your current subscription. 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

인덱스를 사용할 준비가 되면 다음 단계로 이동합니다.

## 마법사 시작

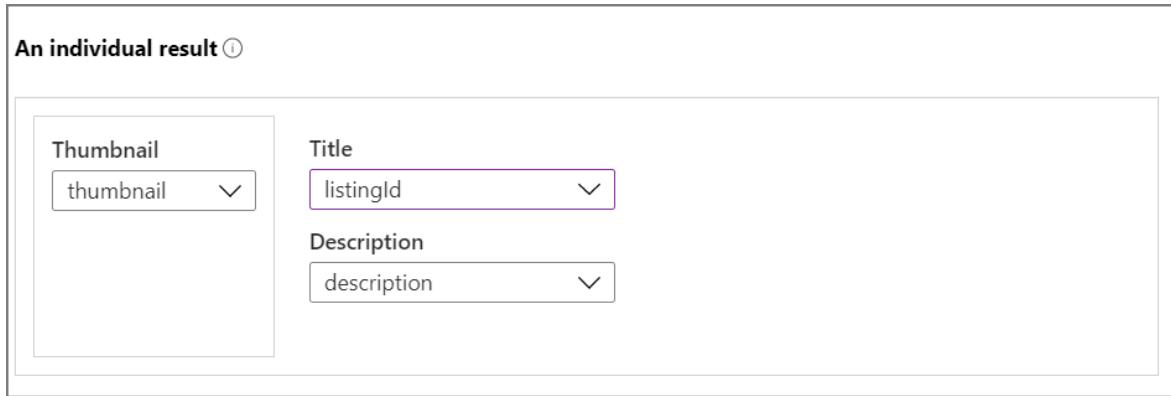
1. Azure 계정을 사용하여 [Azure Portal](#)에 로그인합니다.
2. [검색 서비스를 찾고](#) 개요 페이지의 가운데에 있는 링크에서 **인덱스**를 선택합니다.
3. 기존 인덱스 목록에서 *realestate-us-sample-index*를 선택합니다.
4. 인덱스 페이지의 위쪽에서 **데모 앱 만들기**를 선택하여 마법사를 시작합니다.
5. 첫 번째 마법사 페이지에서 **CORS(원본 간 리소스 공유)** 사용을 선택하여 CORS 지원을 인덱스 정의에 추가합니다. 이 단계는 선택 사항이지만, 이를 수행하지 않으면 로컬 웹앱이 원격 인덱스에 연결되지 않습니다.

## 검색 결과 구성

마법사는 미리 보기 이미지, 제목 및 설명을 위한 공간이 포함된 기본 레이아웃을 렌더링된 검색 결과에 제공합니다. 이러한 각 요소를 지원하는 것은 데이터를 제공하는 인덱스의 필드입니다.

1. [미리 보기]의 *realestate-us-sample* 인덱스에서 *thumbnail* 필드를 선택합니다. 이 샘플은 *thumbnail*이라는 필드에 저장된 URL 주소 이미지 형태의 이미지 미리 보기 를 포함하고 있습니다. 인덱스에 이미지가 없으면 이 필드를 비워 둡니다.
2. [제목]에서 각 문서의 고유성을 나타내는 필드를 선택합니다. 이 샘플에서는 목록 ID 가 적절한 선택입니다.

3. [설명]에서 다른 사용자가 특정 문서를 클릭할지 여부를 결정하는 데 도움이 될 수 있는 세부 정보를 제공하는 필드를 선택합니다.



## 사이드바 추가

검색 서비스는 사이드바로 자주 렌더링되는 패싯 탐색을 지원합니다. 패싯은 인덱스 스키마에서 표시한 대로 필터링 가능 및 패싯 가능 필드를 기반으로 합니다.

Azure AI 검색에서 패싯 탐색은 누적 필터링 환경입니다. 범주 내에서 여러 필터를 선택하면 결과가 확장됩니다(예: 도시 내에서 시애틀 및 벨뷰 선택). 여러 범주에서 여러 필터를 선택하면 결과가 좁혀집니다.

### 💡 팁

전체 인덱스 스키마는 포털에서 볼 수 있습니다. 각 인덱스의 개요 페이지에서 [인덱스 정의\(JSON\)](#) 링크를 찾습니다. 패싯 탐색에 적합한 필드에는 "filterable: true" 및 "facetable: true" 특성이 있습니다.

- 마법사에서 페이지 맨 위에 있는 **사이드바** 탭을 선택합니다. 인덱스에서 필터링 가능하고 패싯 가능한 것으로 특성이 지정된 모든 필드의 목록이 표시됩니다.
- 현재 선택된 패싯 필드를 그대로 적용하고 다음 페이지로 이동합니다.

## 자동 완성 추가

자동 완성 및 쿼리 제안 형식으로 미리 구성된 기능을 사용할 수 있습니다. 마법사는 쿼리 제안을 지원합니다. 사용자가 제공하는 키 입력에 따라 검색 서비스에서 입력으로 선택할 수 있는 "완성된" 쿼리 문자열 목록을 반환합니다.

제안은 특정 필드 정의에서 사용하도록 설정됩니다. 마법사는 제안에 포함된 정보의 양을 구성하는 옵션을 제공합니다.

다음 스크린샷에서는 앱에서 렌더링된 페이지와 함께 마법사의 옵션을 보여 줍니다. 필드 선택이 사용되는 방법 및 "필드 이름 표시"가 제안 내에서 레이블 지정을 포함하거나 제외하는 데 사용되는 방법을 확인할 수 있습니다.

The screenshot shows the 'Create Search App (preview)' interface with the 'Suggestions' tab selected. On the left, there's a preview area and a style section with various options. The main area shows a search suggestion box with the query 'lake wa'. Below it is a table with columns for 'Field name' and 'Show Field Name'. The 'Field name' column contains 'number', 'street', 'city', 'region', and 'postCode'. Each field has a checkbox next to it under 'Show Field Name', all of which are checked. A red box highlights the 'Field name' column, and a red arrow points from it to the search suggestions.

## 제안 추가

제안은 검색 상자에 연결된 자동 쿼리 프롬프트를 나타냅니다. Azure AI 검색은 부분적으로 입력된 검색어의 자동 완성과 잠재적으로 일치하는 문서를 기반으로 하는 드롭다운 목록에 대한 제안이라는 두 가지를 지원합니다.

마법사는 제안을 지원하고 제안된 결과를 제공할 수 있는 필드는 인덱스의 [Suggesters](#) 구성에서 파생됩니다.

JSON

```
"suggesters": [
  {
    "name": "sg",
    "searchMode": "analyzingInfixMatching",
    "sourceFields": [
      "number",
      "street",
      "city",
      "region",
      "postCode",
      "tags"
    ]
  }
]
```

1. 마법사에서 페이지 맨 위에 있는 제안 탭을 선택합니다. 인덱스 스키마에서 제안 공급자로 지정된 모든 필드의 목록이 표시됩니다.
2. 현재 선택 항목을 그대로 적용하고 다음 페이지로 이동합니다.

# 만들기, 다운로드 및 실행

- 페이지 맨 아래에 있는 **데모 앱 만들기**를 선택하여 HTML 파일을 생성합니다.
- 메시지가 표시되면 **앱 다운로드**를 선택하여 파일을 다운로드합니다.
- 파일을 열고 **검색** 단추를 선택합니다. 이 작업은 임의의 결과 집합을 반환하는 빈 쿼리(\*)일 수 있는 쿼리를 실행합니다. 이 페이지는 다음 스크린샷과 유사해야 합니다. 용어를 입력하고 필터를 사용하여 결과를 좁힙니다.

기본 인덱스는 문서 간에 중복된 가상의 생성된 데이터로 구성되며 설명이 이미지와 일치하지 않는 경우도 있습니다. 사용자 고유의 인덱스를 기반으로 하여 앱을 만드는 경우 더 응집력 있는 환경을 사용할 수 있습니다.

The screenshot shows the Azure Search interface with the following details:

- Search Bar:** Lake Washington Boulevard South
- Filter Panel (Left):**
  - type:**  Apartment,  House (249)
  - price:** Range slider from 0.0 to 69 < 1.0m <
  - sqft:** Range slider from 0.0 < 0 < 1.0m <
- Results List (Right):**
  - Result 1:** 9383442  
This is a townhouse and is a short sale. This property has ocean views located close to a river and features a swimming pool, crown mouldings and a large walk in closet.
  - Result 2:** 9382448  
This is a duplex residence and is brand new. This property has ocean views located close to schools and features a swimming pool, beautiful bedroom floors and vaulted ceilings.
  - Result 3:** 9383199  
This is a multi-storey house and is a short sale. Enjoy water frontage located in a culs-de-sac and features top of the line appliances, french doors throughout and a covered front porch.

## 리소스 정리

본인 소유의 구독으로 이 모듈을 진행하고 있는 경우에는 프로젝트가 끝날 때 여기에서 만든 리소스가 계속 필요한지 확인하는 것이 좋습니다. 계속 실행되는 리소스에는 요금이 부과될 수 있습니다. 리소스를 개별적으로 삭제하거나 리소스 그룹을 삭제하여 전체 리소스 세트를 삭제할 수 있습니다.

왼쪽 탐색 창의 **모든 리소스** 또는 **리소스 그룹** 링크를 사용하여 포털에서 리소스를 찾고 관리할 수 있습니다.

무료 서비스에서는 인덱스, 인덱서 및 데이터 원본이 3개로 제한된다는 점을 기억하세요. 포털에서 개별 항목을 삭제하여 제한 이하로 유지할 수 있습니다.

## 다음 단계

데모 앱은 JavaScript 또는 프런트 엔드 코드를 작성하지 않고도 최종 사용자 환경을 시뮬레이트할 수 있지만 자체 프로젝트에서 개념 증명에 가까워지면 실제 단어 앱과 더 가까운 앤드투엔드 코드 샘플 중 하나를 검토하기 때문에 프로토타이핑에 유용합니다.

웹앱에 검색 추가

## 피드백

이 페이지가 도움이 되었나요?

👍 Yes

👎 No

제품 사용자 의견 제공 | Microsoft Q&A에서 도움말 보기

# 빠른 시작: Azure Portal에서 기술 세트 만들기

아티클 • 2024. 10. 16.

이 빠른 시작에서는 Azure AI 검색의 기술 세트가 OCR(광학 인식), 이미지 분석, 언어 감지, 텍스트 번역, 엔터티 인식을 추가하여 검색 인덱스에서 텍스트 검색 가능한 콘텐츠를 생성하는 방법을 알아봅니다.

Azure Portal에서 **데이터 가져오기** 마법사를 실행하여 인덱싱 중에 텍스트 콘텐츠를 생성 및 변환하는 기술을 적용할 수 있습니다. 입력은 원시 데이터(일반적으로 Azure Storage의 Blob)입니다. 출력은 AI에서 생성된 이미지 텍스트, 캡션, 엔터티를 포함하는 검색 가능한 인덱스입니다. 생성된 콘텐츠는 **검색 탐색기**를 사용하여 포털에서 쿼리할 수 있습니다.

준비하려면 마법사를 실행하기 전에 몇 가지 리소스를 만들고 샘플 파일을 업로드합니다.

## 필수 조건

시작하기 전에 다음과 같은 필수 구성 요소를 갖추어야 합니다.

- 활성 구독이 있는 Azure 계정. 체험 계정을 만듭니다.
- Azure AI 검색. [서비스를 만들거나 기존 서비스를 찾습니다](#). 이 빠른 시작에서는 체험 서비스를 사용할 수 있습니다.
- Blob Storage가 있는 Azure Storage 계정.

### ① 참고

이 빠른 시작에서는 [Azure AI 서비스](#)를 사용하여 AI 변환을 수행합니다. 워크로드가 너무 작으므로 Azure AI 서비스는 최대 20개의 트랜잭션을 무료로 처리하기 위해 백그라운드에서 탭으로 처리됩니다. Azure AI 다중 서비스 리소스를 만들지 않고도 이 연습을 완료할 수 있습니다.

## 데이터를 설정합니다.

다음 단계에서는 다른 유형의 콘텐츠 파일을 저장할 Azure Storage의 blob 컨테이너를 설정합니다.

- 여러 종류의 작은 파일 집합으로 구성된 [샘플 데이터를 다운로드](#)하세요.

2. Azure 계정을 사용하여 [Azure Portal](#) 에 로그인합니다.

3. [Azure Storage 계정을 만들거나 기존 계정을 찾습니다](#).

- 대역폭 요금이 부과되지 않도록 Azure AI 검색과 동일한 지역을 선택합니다.
- StorageV2(범용 V2)를 선택합니다.

4. Azure Portal에서 Azure Storage 페이지를 열고 컨테이너를 만듭니다. 기본 액세스 수준을 사용할 수 있습니다.

5. 컨테이너에서 **업로드**를 선택하여 샘플 파일을 업로드합니다. 전체 텍스트를 해당 네이티브 형식으로 검색할 수 없는 이미지와 애플리케이션 파일을 포함하여 다양한 콘텐츠 형식이 있습니다.

NAME	LAST MODIFIED	BLOB TYPE	CONTENT TYPE
10-K-FY16.html	2018-04-02T21:10:35.000Z	Block Blob	text/html
5074.clip_image002_6FE27E85.png	2018-04-02T21:10:39.000Z	Block Blob	image/png
Cognitive Search and Content Intelligence.pptx	2018-04-02T21:10:50.000Z	Block Blob	application/vnd.openxmlformats-officedocument.presentationml.presentation
Cognitive Services and Bots (spanish).pdf	2018-04-02T21:10:41.000Z	Block Blob	application/pdf
guthrie.jpg	2018-04-02T21:10:23.000Z	Block Blob	image/jpeg
MSFT_cloud_architecture_contoso.pdf	2018-04-02T21:10:35.000Z	Block Blob	application/pdf
MSFT_FY17_10K.docx	2018-04-02T21:10:36.000Z	Block Blob	application/vnd.openxmlformats-officedocument.wordprocessingml.document
NYSE_LNKD_2015.PDF	2018-04-02T21:10:35.000Z	Block Blob	application/pdf
redshirt.jpg	2018-04-02T21:10:31.000Z	Block Blob	image/jpeg
satyanadellalinux.jpg	2018-04-02T21:10:33.000Z	Block Blob	image/jpeg
satyasletter.txt	2018-04-02T21:10:22.000Z	Block Blob	text/plain

이제 데이터 가져오기 마법사로 이동할 준비가 되었습니다.

## 데이터 가져오기 마법사 실행

1. Azure 계정을 사용하여 [Azure Portal](#) 에 로그인합니다.

2. [검색 서비스를 찾고](#) 개요 페이지의 명령 모음에서 **데이터 가져오기**를 선택하여 네 단계로 검색 가능한 콘텐츠를 만듭니다.



## 1단계: 데이터 소스 만들기

1. 데이터에 연결에서 Azure Blob 스토리지를 선택합니다.

2. 사용자가 만든 스토리지 계정에 대한 기존 연결을 선택하고 생성한 컨테이너를 선택합니다. 데이터 원본의 이름을 지정하고, 나머지는 기본값을 사용합니다.

## Import data ...

X

## \* 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

Azure Blob Storage

## Data source name \*

cog-search-demo-ds

## Data to extract ⓘ

Content and metadata

## Parsing mode

Default

## Connection string \*

DefaultEndpointsProtocol=https;AccountName= ...

[Choose an existing connection](#)

## Managed identity authentication ⓘ

 None System-assigned User-assigned

## Container name \* ⓘ

cog-search-demo

## Blob folder ⓘ

your/folder/here

## Description

(optional)

다음 페이지를 계속합니다.

"데이터 원본에서 인덱스 스키마 검색 오류"가 발생하면 마법사를 구동하는 인덱서는 데이터 원본에 연결할 수 없습니다. 대부분의 경우 데이터 원본에는 보안 보호가 있습니다. 다음 솔루션을 시도한 다음, 마법사를 다시 실행합니다.

[+] 테이블 확장

보안 기능	솔루션
리소스를 사용하려면 Azure 역할이 필요하거나 해당 액세스 키를 사용하지 않도록 설정해야 함	<a href="#">신뢰할 수 있는 서비스로 연결 또는 관리 ID를 사용하여 연결</a>
리소스가 IP 방화벽 뒤에 있음	<a href="#">검색 및 Azure Portal에 대한 인바운드 규칙을 만듦</a>
리소스에는 프라이빗 엔드포인트 연결이 필요함	<a href="#">프라이빗 엔드포인트에 연결</a>

## 2단계: 인식 기술 추가

다음으로, OCR, 이미지 분석 및 자연어 처리를 호출하도록 AI 보강을 구성합니다.

1. 이 빠른 시작에서는 **무료** Azure AI 서비스 리소스를 사용합니다. 샘플 데이터는 14 개의 파일로 구성되어 있으므로 이 빠른 시작에서는 Azure AI 서비스에서 20개의 트랜잭션을 무료로 할당하는 것으로 충분합니다.

Dashboard > srch-demo-westus2 >

## Import data

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

**⚠️** 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

To power your cognitive skills, select an existing Cognitive Services resource or create a new one. The Cognitive Services resource should be in the same region as your Azure Cognitive Search service. The execution of cognitive skills will be billed to the selected resource. Otherwise, the number of enrichments executions will be limited. [Learn more](#)

Refresh

Cognitive Services Resource Name	Region
Free (Limited enrichments)	

Create new Cognitive Services resource

2. **보강 추가**를 확장하고 4개를 선택합니다.

마법사 페이지에 이미지 분석 기술을 추가하려면 OCR을 사용하도록 설정합니다.

엔터티 인식(사람, 조직 및 위치)과 이미지 분석 기술(태그, 캡션)을 선택합니다.

✓ 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 \* ⓘ  
cog-search-demo-ss ✓

Enable OCR and merge all text into **merged\_content** field ⓘ

Source data field \*  
merged\_content ✓

Enrichment granularity level ⓘ  
Source field (default) ✓

Enable incremental enrichment ⓘ

Checked items below require a field name.

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 type="checkbox"/> Extract key phrases		keyphrases
<input type="checkbox"/> Detect language		language
<input type="checkbox"/> Translate text	Target Language English	translated_text
<input type="checkbox"/> Extract personally identifiable...		pii_entities

Image Cognitive Skills	Field name
<input checked="" type="checkbox"/> Generate tags from images	imageTags ✓
<input checked="" type="checkbox"/> Generate captions from images	imageCaption
<input type="checkbox"/> Identify celebrities from images	imageCelebrities

다음 페이지를 계속합니다.

## 3단계: 인덱스 구성

인덱스는 검색 가능한 콘텐츠를 포함하고 있으며, **데이터 가져오기** 마법사는 일반적으로 데이터 원본을 샘플링하여 스키마를 만들 수 있습니다. 이 단계에서는 생성된 스키마를 검토하고, 잠재적으로 설정을 수정할 수 있습니다.

이 빠른 시작에서 마법사는 기본값을 적절하게 설정합니다.

- 기본 필드는 기존 Blob의 데이터 속성과 보강 출력을 포함하는 새 필드(예: `people`, `organizations`, `locations`)를 기반으로 합니다. 데이터 형식은 메타데이터 및 데이

터 샘플링에서 유추됩니다.

- 기본 문서 키는 *metadata\_storage\_path*입니다(필드에 고유 값이 포함되어 있으므로 선택됨).
- 기본 특성은 **조회 가능**하고 **검색 가능**합니다. **검색 가능**은 필드의 전체 텍스트 검색을 허용합니다. **조회 가능**은 필드 값을 결과에 반환할 수 있다는 뜻입니다. 마법사는 사용자가 기술 세트를 통해 필드를 생성했기 때문에 이 필드를 조회 및 검색 가능하도록 하려고 한다고 가정합니다. 필터 식에서 필드를 사용하려면 **필터링 가능**을 선택합니다.

Field name	Type	Retrievable	Filterable	Sortable	Facetable	Searchable	Analyzer
content	Edm.String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Standard - Luce...
metadata_storage_content_type	Edm.String	<input type="checkbox"/>					
metadata_storage_size	Edm.Int64	<input type="checkbox"/>					
metadata_storage_last_modified	Edm.DateTi...	<input type="checkbox"/>					
metadata_storage_content_md5	Edm.String	<input type="checkbox"/>					
metadata_storage_name	Edm.String	<input type="checkbox"/>					
metadata_storage_path	Edm.String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
metadata_storage_file_extension	Edm.String	<input type="checkbox"/>					
metadata_content_type	Edm.String	<input type="checkbox"/>					
people	Collection(E...)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Standard - Luce...
organizations	Collectio...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Standard - Luce...
locations	Collection(E...)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Standard - Luce...
merged_content	Edm.String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Standard - Luce...
text	Collection(E...)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Standard - Luce...
layoutText	Collection(E...)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Standard - Luce...

필드를 **조회 가능**으로 표시한다고 해서 필드가 검색 결과에 반드시 나타나야 하는 것은 아닙니다. **select** 쿼리 매개 변수로 포함할 필드를 지정하여 검색 결과의 구성을 제어할 수 있습니다.

다음 페이지를 계속합니다.

## 4단계: 인덱서 구성

인덱서는 인덱싱 프로세스를 구동합니다. 인덱서는 데이터 원본 이름, 대상 인덱스 및 실행 빈도를 지정합니다. **데이터 가져오기** 마법사는 여러 개체를 만들고, 다시 설정하고 반복적으로 실행할 수 있는 인덱서를 포함합니다.

1. 인덱서 페이지에서 기본 이름을 적용하고, **한 번**을 선택합니다.

Dashboard > srch-demo-westus2 >

## Import data

\* Connect to your data    Add cognitive skills (Optional)    \* Customize target index    [Create an indexer](#)

**Indexer**

Name \*  ✓

Schedule ⓘ  Once   Hourly   Daily   Custom

Description

[Advanced options](#)

2. 제출을 선택하여 인덱서를 만드는 동시에 실행합니다.

## 상태 모니터링

왼쪽 탐색 창에서 **인덱서**를 선택하여 상태를 모니터링한 다음, 인덱서를 선택합니다. 기술 기반 인덱싱은 텍스트 기반 인덱싱, 특히 OCR 및 이미지 분석보다 오래 걸립니다.

Dashboard > srch-demo-westus2 >

## cog-search-demo

Indexer

Run Reset Save Refresh Delete

Execution history    Settings    Indexer Definition (JSON)

Number of recent runs to show: 5  
20s

15s  
10s  
5s  
0s

06/01

Status    Last run    Duration    Docs succeeded    Errors/Warnings

Status	Last run	Duration	Docs succeeded	Errors/Warnings
<span style="color: green;">✓ Success</span>	6/1/2022, 14:38:07	16.27 s	14/14	0/1

✓ Success

✗ Failed



실행 상태에 대한 세부 정보를 확인하려면 **성공(또는 실패)**을 선택하여 실행 세부 정보를 봅니다.

이 데모에는 다음과 같은 몇 가지 경고가 있습니다. "Could not execute skill because one or more skill input was invalid." 이러한 경고는 데이터 원본의 PNG 파일이 엔터티 인식에 텍스트 입력을 제공하지 않는다는 것을 알려줍니다. 이 경고는 업스트림 OCR 기술이 이미지의 텍스트를 인식하지 못하여 다운스트림 엔터티 인식 기술에 텍스트 입력을 제공할 수 없기 때문에 발생합니다.

경고는 기술 세트 실행에서 일반적입니다. 데이터가 반복되는 기술에 익숙해지면 패턴을 발견하고 무시해도 안전한 경고를 알아볼 수 있습니다.

## Search 탐색기에서 쿼리

인덱스가 생성된 후에는 **검색 탐색기**를 사용하여 결과를 반환할 수 있습니다.

1. 왼쪽에서 **인덱스**를 선택한 다음, **인덱스**를 선택합니다. **검색 탐색기**는 첫 번째 탭에 있습니다.
2. `satya nadella`처럼 인덱스를 쿼리하는 검색 문자열을 입력합니다. 검색창에는 키워드, 인용문으로 묶인 구 및 연산자(`"Satya Nadella" + "Bill Gates" + "Steve Ballmer"`)를 입력할 수 있습니다.

결과는 자세한 JSON으로 반환되어 특히 대용량 문서에서 읽기 어려울 수 있습니다. 이 도구를 검색하는 몇 가지 팁은 다음과 같습니다.

- 결과를 형성하는 매개 변수를 지정하려면 JSON 보기로 전환합니다.
- 결과에서 필드를 제한하려면 `select`를 추가합니다.
- 일치 항목 수를 표시하려면 `count`를 추가합니다.
- JSON 내에서 특정 속성 또는 용어를 검색하려면 CTRL-F를 사용합니다.

보기예 붙여넣을 수 있는 몇 가지 JSON은 다음과 같습니다.

```
JSON

{
  "search": "\"Satya Nadella\" +\"Bill Gates\" +\"Steve Ballmer\"",
  "count": true,
  "select": "content, people"
}
```

1

쿼리 문자열은 대/소문자를 구분하므로 "알 수 없는 필드" 메시지가 표시되면 필드 또는 인덱스 정의(JSON)를 검사하여 이름과 대/소문자를 확인합니다.

# 핵심 내용

이제 첫 번째 기술 세트를 만들고 기술 기반 인덱싱의 기본 단계를 알아보았습니다.

여러분이 기억했으면 하는 주요 개념 중 하나는 종속성입니다. 기술 세트는 인덱서에 바인딩되며, 인덱서는 Azure 및 원본 전용입니다. 이 빠른 시작에서는 Azure Blob Storage를

사용하지만 다른 Azure 데이터 원본도 가능합니다. 자세한 내용은 [Azure AI 검색의 인덱서](#)를 참조하세요.

또 다른 중요한 개념은 기술이 콘텐츠 형식에 대해 작동하며, 다른 형식의 콘텐츠를 사용하는 경우 일부 입력이 생략될 수 있다는 것입니다. 또한 많은 파일 또는 필드가 서비스 계층의 인덱서 한도를 초과할 수 있습니다. 이러한 이벤트가 발생할 때 경고가 표시되는 것이 일반적입니다.

출력은 검색 인덱스로 라우팅되고 인덱싱 중에 만들어진 이름-값 쌍과 인덱스의 개별 필드 간에 매핑됩니다. 내부적으로, 마법사는 [보강 트리](#)를 설정하고 [기술 세트](#)를 정의하여 작업 및 일반적인 흐름의 순서를 설정합니다. 이러한 단계는 마법사에서 숨겨지지만, 코드 작성을 시작할 때에는 해당 개념이 중요합니다.

마지막으로, 인덱스를 쿼리하여 내용을 확인할 수 있는 방법을 알아보았습니다. 결국 Azure AI 검색은 검색 가능한 인덱스를 제공하며, 이 인덱스는 [단순](#) 또는 [완전히 확장된 쿼리 구문](#)을 사용하여 쿼리할 수 있습니다. 보강된 필드를 포함하는 인덱스는 다른 인덱스와 비슷합니다. 원하는 경우 표준 또는 [사용자 지정 분석기](#), [점수 매기기 프로필](#), [동의어](#), [패싯 탐색](#), 지역 검색, 기타 Azure AI 검색 기능을 통합할 수 있습니다.

## 리소스 정리

본인 소유의 구독으로 이 모듈을 진행하고 있는 경우에는 프로젝트가 끝날 때 여기에서 만든 리소스가 계속 필요한지 확인하는 것이 좋습니다. 계속 실행되는 리소스에는 요금이 부과될 수 있습니다. 리소스를 개별적으로 삭제하거나 리소스 그룹을 삭제하여 전체 리소스 세트를 삭제할 수 있습니다.

왼쪽 탐색 창의 **모든 리소스** 또는 **리소스 그룹** 링크를 사용하여 포털에서 리소스를 찾고 관리할 수 있습니다.

무료 서비스를 사용한 경우 세 개의 인덱스, 인덱서 및 데이터 원본으로 제한됩니다. 포털에서 개별 항목을 삭제하여 제한 이하로 유지할 수 있습니다.

## 다음 단계

포털, .NET SDK 또는 REST API를 사용하여 기술 세트를 만들 수 있습니다. 지식을 더 자세히 알아보려면 REST 클라이언트 및 더 많은 샘플 데이터를 사용하여 REST API를 사용해 보세요.

**자습서: REST API를 사용하여 JSON Blob에서 텍스트 및 구조 추출**

# 피드백

이 페이지가 도움이 되었나요?

Yes

No

[제품 사용자 의견 제공](#) | Microsoft Q&A에서 도움말 보기

# 빠른 시작: Azure Portal에서 지식 저장소 만들기

아티클 • 2024. 09. 03.

이 빠른 시작에서는 Azure AI 검색의 [AI 보강 파이프라인](#)에서 생성된 출력의 리포지토리 역할을 하는 [지식 저장소](#)를 만듭니다. 지식 저장소를 사용하면 Azure Storage에서 생성된 콘텐츠를 검색 이외의 워크로드에 사용할 수 있습니다.

먼저 Azure Storage에서 몇 가지 샘플 데이터를 설정합니다. 그런 다음, [데이터 가져오기](#) 마법사를 실행하여 지식 저장소도 만드는 보강 파이프라인을 만듭니다. 지식 저장소에는 데이터 원본에서 가져온 원본 콘텐츠(호텔에 대한 고객 검토)와 감정 레이블, 핵심 구 추출 및 영어가 아닌 고객 의견의 텍스트 번역이 포함된 AI 생성 콘텐츠가 포함됩니다.

## 필수 조건

시작하기 전에 다음과 같은 필수 구성 요소를 갖추어야 합니다.

- 활성 구독이 있는 Azure 계정. [체험 계정을 만들거나 찾습니다](#).
- Azure AI 검색. 계정에서 [서비스를 만들거나 기존 서비스를 찾습니다](#). 이 빠른 시작에서는 체험 서비스를 사용할 수 있습니다.
- Azure Storage. [계정을 만들거나 기존 스토리지 계정을 찾습니다](#). 계정 유형은 **StorageV2(범용 V2)**여야 합니다.
- Azure Storage에서 호스트되는 샘플 데이터:

[HotelReviews\\_Free.csv를 다운로드합니다](#). 이 CSV에는 단일 호텔에 대한 19개의 고객 피드백이 포함되어 있습니다(Kaggle.com에서 가져옴). 파일은 다른 샘플 데이터와 함께 리포지토리에 있습니다. 전체 리포지토리를 사용하지 않으려면 원시 콘텐츠를 복사하여 디바이스의 스프레드시트 앱에 붙여넣습니다.

Azure Storage의 [BLOB 컨테이너에 파일을 업로드](#)합니다.

또한 이 빠른 시작에서는 AI 보강을 위해 [Azure AI 서비스](#)를 사용합니다. 워크로드가 너무 작으므로 Azure AI 서비스는 최대 20개의 트랜잭션을 무료로 처리하기 위해 백그라운드에서 탭으로 처리됩니다. 즉, 추가 Azure AI 다중 서비스 리소스를 만들지 않고도 이 연습을 완료할 수 있습니다.

## 마법사 시작

1. Azure 계정을 사용하여 [Azure Portal](#) 에 로그인합니다.
2. [Search Service 찾기](#) 이후 개요 페이지의 명령 모음에서 **데이터 가져오기**를 선택하여 다음 네 단계에 따라 지식 저장소를 만듭니다.



## 1단계: 데이터 소스 만들기

데이터는 한 CSV 파일에 있는 여러 행이므로 각 행에 대해 하나의 검색 문서를 가져오도록 구문 분석 모드를 설정합니다.

1. 데이터에 연결에서 Azure Blob 스토리지를 선택합니다.
2. 이름에 "hotel-reviews-ds"를 입력합니다.
3. 추출할 데이터로 콘텐츠 및 메타데이터를 선택합니다.
4. 구문 분석 모드에 대해 **분리된 텍스트**를 선택한 다음, 첫 줄에 헤더 포함 확인란을 선택합니다. 구분 기호 문자가 쉼표(,)인지 확인합니다.
5. 스토리지 계정이 동일한 구독에 있는 경우 **연결 문자열**에서 기존 연결을 선택합니다. 그렇지 않으면 연결 문자열을 Azure Storage 계정에 붙여넣습니다.

연결 문자열은 모든 권한이 가능하며 형식은 다음과 같습니다.

```
DefaultEndpointsProtocol=https;AccountName=<YOUR-ACCOUNT-NAME>;AccountKey=<YOUR-ACCOUNT-KEY>;EndpointSuffix=core.windows.net
```

또는 연결 문자열은 Azure Storage에서 [역할이 구성 및 할당](#)되었다고 가정할 때 관리 ID를 참조할 수 있습니다.

```
ResourceId=/subscriptions/{YOUR-SUBSCRIPTION-ID}/resourceGroups/{YOUR-RESOURCE-GROUP-NAME}/providers/Microsoft.Storage/storageAccounts/{YOUR-ACCOUNT-NAME};
```

6. 컨테이너에서 데이터를 보유하는 Blob 컨테이너의 이름("hotel-reviews")을 입력합니다.

이 페이지는 다음 스크린샷과 비슷합니다.

## Import data

\* Connect to your data Add cognitive search (Optional) Customize target index Create an indexer

Create and load a search index using data from an existing Azure data source in your current subscription. Azure 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

Azure Blob Storage

\* Name

hotel-reviews-ds

Data to extract

Content and metadata

Parsing mode

Delimited text

First Line Contains Header



Delimiter Character

,

\* Connection string

DefaultEndpointsProtocol=https;AccountName=mydemoblobstore;...



[Choose an existing connection](#)

\* Container name

hotel-reviews

[Next: Add cognitive search \(Optional\)](#)

7. 다음 페이지를 계속합니다.

## 2단계: 기술 추가

이 마법사 단계에서는 AI 보강 기술을 추가합니다. 원본 데이터는 영어와 프랑스어로 된 고객 리뷰로 구성됩니다. 이 데이터 세트와 관련된 기술에는 핵심 구 추출, 감정 감지 및 텍스트 번역이 포함됩니다. 이후 단계에서 이러한 보강은 지식 저장소에 Azure 테이블로 "프로젝션"됩니다.

1. **Azure AI 서비스 연결**을 확장합니다. **무료(제한적 보강)**가 기본적으로 선택되어 있습니다. HotelReviews-Free.csv의 레코드 수가 19개이고 이 무료 리소스에서 하루 최대 20개의 트랜잭션을 허용하므로 이 리소스를 사용할 수 있습니다.
2. **보강 추가**를 확장합니다.
3. **기술 세트 이름**으로 "hotel-reviews-ss"를 입력합니다.
4. **원본 데이터 필드**에 대해 reviews\_text를 선택합니다.
5. **보강 세분화 수준**에 대해 **페이지(5,000자 청크)**를 선택합니다.
6. **텍스트 인지 기술**에 대해 다음 기술을 선택합니다.
  - **핵심 구 추출**
  - **텍스트 번역**

- 언어 감지
- 감정 감지

페이지는 다음 스크린샷과 같습니다.

Text Cognitive Skills	Parameter	Field name
<input type="checkbox"/> Extract people names		people
<input type="checkbox"/> Extract organization names		organizations
<input 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 type="checkbox"/> Extract personally identifiable infor...		pii_entities
<input checked="" type="checkbox"/> Detect sentiment		sentiment

7. 아래로 스크롤하여 **지식 저장소에 보강 저장**을 펼칩니다.

8. **기준 연결 선택**을 선택한 다음, Azure Storage 계정을 선택합니다. 컨테이너 페이지가 표시되므로 프로젝션을 위한 컨테이너를 만들 수 있습니다. 원본 콘텐츠와 기술 자료 저장소 콘텐츠 간을 구분하기 위해 "kstore-hotel-reviews"와 같은 접두사 명명 규칙을 채택하는 것이 좋습니다.

9. 데이터 가져오기 마법사로 돌아가서 다음 **Azure 테이블 프로젝션**을 선택합니다. 마법사는 항상 **문서** 프로젝션을 제공합니다. 선택한 기술(예: **핵심 구**) 또는 보강 세분화(**페이지**)에 따라 다른 프로젝션이 제공됩니다.

- 문서
- 페이지
- 핵심 구

다음 스크린샷은 마법사의 테이블 프로젝션 선택 항목을 보여 줍니다.

Import data ... X

---

- ▽ Attach Cognitive Services
- ▽ Add enrichments
- △ Save enrichments to a knowledge store

A knowledge store allows you to project your enriched documents into tables and blobs. [Learn more about Knowledge Store](#)

Storage account connection string \*

DefaultEndpointsProtocol=https;AccountName=blobstorage2;AccountKey=aaabbccdddee1122233444555==;EndpointS✓

Choose an existing connection

Azure table projections

- Documents
- Pages
- Key phrases

Azure blob projections

- Document

10. 다음 페이지를 계속합니다.

## 3단계: 인덱스 구성

이 마법사 단계에서는 선택적 전체 텍스트 검색 쿼리에 대한 인덱스를 구성합니다. 지식 저장소에는 검색 인덱스가 필요하지 않지만 인덱서를 실행하려면 검색 인덱스가 필요합니다.

이 단계에서 마법사는 데이터 원본을 샘플링하여 필드와 데이터 형식을 유추합니다. 원하는 동작의 특성만 선택하면 됩니다. 예를 들어 **Retrievable** 특성을 사용하면 검색 서비스에서 필드 값을 반환할 수 있지만, **Searchable** 특성을 사용하면 필드에서 전체 텍스트를 검색할 수 있습니다.

1. 인덱스 이름으로 "hotel-reviews-idx"를 입력합니다.
2. 특성의 경우 파이프라인에서 만드는 새 필드에 대해 **Retrievable** 및 **Searchable**을 적용합니다.

인덱스는 다음 이미지와 비슷합니다. 긴 목록이므로 일부 필드가 이미지에 표시되지 않습니다.

Import data

Add field Add subfield Delete

FIELD NAME	TYPE	RETRIEVABLE	FILTERABLE	SORTABLE	FACTABLE	SEARCHABLE	ANALYZER
keyphrases	Collection(E...)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Standard - Luce... <input type="button" value="▼"/>
translated_text	Collection(E...)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	English - Lucene <input type="button" value="▼"/>
sentiment	Collection(E...)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>			

Previous: Add cognitive search (Optional) Next: Create an indexer

3. 다음 페이지를 계속합니다.

## 4단계: 인덱서 구성 및 실행

이 마법사 단계에서는 이전 마법사 단계에서 정의한 데이터 원본, 기술 세트 및 인덱스를 모두 가져오는 인덱서를 구성합니다.

1. 이름으로 "hotel-reviews-idxr"을 입력합니다.
2. 일정에 대해 한 번(기본값)을 유지합니다.
3. 제출을 선택하여 인덱서를 실행합니다. 데이터 추출, 인덱싱, 인지 기술 적용은 모두 이 단계에서 수행됩니다.

## 5단계: 상태 확인

개요 페이지에서 페이지 중간에 있는 인덱서 탭을 연 다음, **hotels-reviews-idxr**을 선택합니다. 상태가 1~2분 내에 오류 및 경고 없이 "진행 중"에서 "성공"으로 진행되어야 합니다.

## Azure Portal에서 테이블 확인

1. Azure Portal에서 지식 저장소를 만드는데 사용된 [스토리지 계정을 엽니다](#).
2. 스토리지 계정의 왼쪽 탐색 창에서 **스토리지 브라우저(미리 보기)**를 선택하여 새 테이블을 봅니다.

"보강 추가" 페이지의 "보강 저장" 섹션에 제공된 각 프로젝션에 대해 하나씩 세 개의 테이블이 표시됩니다.

- "hotelReviewssDocuments"는 컬렉션이 아닌 문서 보강 트리의 모든 첫 번째 수준 노드를 포함합니다.
- "hotelReviewssKeyPhrases"는 모든 리뷰에서 추출한 핵심 구의 긴 목록을 포함합니다. 핵심 구 및 엔터티와 같은 컬렉션(배열)을 출력하는 기술은 출력을 독

립형 테이블로 보냅니다.

- "hotelReviewssPages"는 문서에서 분할된 각 페이지에 대해 만들어진 보강 필드를 포함합니다. 이 기술 세트 및 데이터 원본에서는 감정 레이블 및 번역된 텍스트로 구성된 페이지 수준 보강이 있습니다. 기술 세트 정의에서 "페이지" 세분성을 선택하는 경우 페이지 테이블(또는 특정 세분성 수준을 지정하는 경우 문장 테이블)이 만들어집니다.

이러한 테이블은 모두 다른 도구 및 앱에서 테이블 관계를 지원하기 위한 ID 열을 포함합니다. 테이블을 열 때 이러한 필드를 지나면서 스크롤하여 파이프라인에서 추가한 콘텐츠 필드를 봅니다.

이 빠른 시작에서 "hotelReviewssPages"에 대한 테이블은 다음 스크린샷과 비슷해야 합니다.

The screenshot shows the Azure Storage browser (preview) interface. On the left, there's a navigation sidebar with options like blobstorage2, Favorites, Recently viewed, Blob containers, File shares, Queues, and Tables. Under Tables, 'hotelReviewsPages' is selected. The main area shows the table structure with columns: Pagesid, languageCode, sentimentScore, and translatedText. Three rows of data are visible:

Pagesid	languageCode	sentimentScore	translatedText
10 min walk al...	en	mixed	Pleasant 10 min walk al...
with very fri...	en	positive	Nice hotel , with very fri...
10+ min walk to...	en	positive	It was a 10 min+ walk to...

## 정리

본인 소유의 구독으로 이 모듈을 진행하고 있는 경우에는 프로젝트가 끝날 때 여기에서 만든 리소스가 계속 필요한지 확인하는 것이 좋습니다. 계속 실행되는 리소스에는 요금이 부과될 수 있습니다. 리소스를 개별적으로 삭제하거나 리소스 그룹을 삭제하여 전체 리소스 세트를 삭제할 수 있습니다.

왼쪽 탐색 창의 **모든 리소스** 또는 **리소스 그룹** 링크를 사용하여 포털에서 리소스를 찾고 관리할 수 있습니다.

무료 서비스를 사용하는 경우 인덱스, 인덱서, 데이터 원본 3개로 제한됩니다. 포털에서 개별 항목을 삭제하여 제한 이하로 유지할 수 있습니다.

💡 팁

이 연습을 반복하거나 다른 AI 보강 연습을 시도하려면 `hotel-reviews-idxr` 인덱서 및 관련 개체를 삭제하여 다시 만드세요. 인덱서를 삭제하면 사용 가능한 일별 트랜잭션 카운터가 0으로 다시 설정됩니다.

## 다음 단계

이제 지식 저장소를 소개했으므로 REST API 연습으로 전환하여 각 단계를 자세히 살펴봅니다. 마법사에서 내부적으로 처리한 작업은 REST 연습에서 설명합니다.

[REST를 사용하여 지식 저장소 만들기](#)

## 피드백

이 페이지가 도움이 되었나요?

 Yes

 No

[제품 사용자 의견 제공](#) | [Microsoft Q&A에서 도움말 보기](#)

# 빠른 시작: 검색 탐색기를 사용하여 Azure Portal에서 쿼리 실행

아티클 • 2024. 09. 03.

이 빠른 시작에서는 Azure AI 검색에서 검색 인덱스에 대해 쿼리를 실행하는 데 사용되는 Azure Portal의 기본 제공 쿼리 도구인 **검색 탐색기**를 사용하는 방법을 알아봅니다. 쿼리 또는 필터 식을 테스트하거나 인덱스에 콘텐츠가 있는지 확인하는 데 사용합니다.

이 빠른 시작에서는 기존 인덱스를 사용하여 검색 탐색기를 보여줍니다.

## 💡 팁

이제 검색 탐색기가 이미지 검색을 지원합니다. [빠른 시작: Azure Portal의 이미지 검색](#)에서 단계를 제공합니다.

## 필수 조건

시작하기 전에 다음과 같은 필수 구성 요소를 갖추어야 합니다.

- 활성 구독이 있는 Azure 계정. [체험 계정을 만듭니다](#).
- Azure AI 검색 서비스. [서비스를 만들거나](#) 현재 구독에서 [기존 서비스를 찾습니다](#). 이 빠른 시작에서는 체험 서비스를 사용할 수 있습니다.
- realestate-us-sample-index*는 이 빠른 시작에 사용됩니다. 인덱스를 만들려면 [데이터 가져오기 마법사](#)를 사용하고, 기본 제공 샘플 데이터를 선택하고, 모든 기본값을 사용하여 마법사를 단계별로 실행합니다.

The screenshot shows the Microsoft Azure portal interface. At the top, there's a blue header bar with the Microsoft Azure logo, a search bar containing 'Search resources, services, and docs (G+/-)', and a three-dot menu icon. Below the header, the URL 'Dashboard > demo-search-svc >' is visible. The main content area has a title 'Import data' with a close button ('X'). Underneath, there are several tabs: 'Connect to your data' (which is underlined), 'Add cognitive skills (Optional)', 'Customize target index', and 'Create an indexer'. A descriptive text block says: '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.' It includes a 'Learn more' link. Below this, there are sections for 'Data Source' (set to 'Samples'), 'Type' (with two options: 'SQL' and 'Azure Cosmos DB'), and 'Name' (with two entries: 'realestate-us-sample' and 'hotels-sample').

# 검색 탐색기 시작

1. Azure Portal [의 대시보드에서 검색 개요 페이지를 열거나 서비스를 찾습니다](#).
2. 다음과 같이 명령 모음에서 검색 탐색기를 엽니다.



또는 다음과 같이 열려 있는 인덱스에서 포함된 검색 탐색기 탭을 사용합니다.

## 쿼리의 두 가지 방법

검색 탐색기에서 쿼리하는 방법에는 두 가지가 있습니다.

- 쿼리 뷰는 기본 검색 창을 제공합니다. 부울이 있는 자유 텍스트 쿼리 또는 빈 쿼리를 허용합니다. 예들 들어 `seattle condo +parking` 입니다.
- JSON 뷰는 매개 변수화된 쿼리를 지원합니다. 필터, orderby, select, count, searchFields 및 기타 모든 매개 변수는 JSON 보기에서 설정해야 합니다.

### 팁

JSON 뷰는 매개 변수 이름을 완성하기 위한 intellisense를 제공합니다. JSON 보기 안에 커서를 놓고 공백 문자를 입력하여 모든 쿼리 매개 변수 목록을 표시하거나 "s"와 같은 하나의 문자를 입력하여 "s"로 시작하는 쿼리 매개 변수만 표시합니다. Intellisense는 유효하지 않은 매개 변수를 제외하지 않으므로 사용자가 최선의 판단을 내립니다.

매개 변수화된 쿼리에 대해 **JSON 보기**로 전환합니다. 이 문서의 예제에서는 전체적으로 JSON 보기로 가정합니다. 이 문서의 JSON 예제를 텍스트 영역에 붙여넣을 수 있습니다.

The screenshot shows the Azure portal interface for managing a search service. At the top, there's a navigation bar with 'Microsoft Azure ...' and a search bar. Below the navigation bar, the path 'Home > free-demo-search-svc | Indexes >' is displayed. The main title is 'realestate-us-sample-index'. On the left, there are buttons for 'Save', 'Discard', 'Refresh', 'Create Demo App', 'Edit JSON', and 'Delete'. Below these buttons, there are two sections: 'Documents' (4,959) and 'Storage' (12.21 MB). Under the 'Search explorer' tab, there are links for 'Fields', 'CORS', 'Scoring profiles', and 'Vector profiles'. A dropdown menu shows the date '2023-10-01-Preview' and a 'View' option. In the bottom right corner, there are two buttons: 'Query view' and 'JSON view', with 'JSON view' being highlighted by a red box. The JSON query editor contains the following code:

```
1 {  
2   "search": "*"  
3 }
```

A large blue button at the bottom right says 'Search'.

# 지정되지 않은 쿼리 실행

검색 탐색기에서 POST 요청은 [검색 POST REST API](#)를 사용하여 내부적으로 공식화되며 응답은 자세한 JSON 문서로 반환됩니다.

콘텐츠를 대략적으로 살펴보려면 용어를 입력하지 않은 상태로 검색을 클릭하여 빈 검색을 실행합니다. 빈 검색은 전체 문서를 반환하여 문서 컴퍼지션을 검토할 수 있도록 하므로 첫 번째 쿼리로서 유용합니다. 빈 검색에서는 검색 점수가 없으며 문서가 임의 순서로 반환됩니다(모든 문서에 대한 "@search.score": 1). 기본적으로 하나의 검색 요청에서 50개의 문서가 반환됩니다.

빈 검색에 해당하는 구문은 \* 또는 "search": "\*" 입니다.

```
JSON

{
  "search": "*"
}
```

결과

## realestate-us-sample-index ...

X

Search explorer Fields CORS Scoring profiles Vector profiles

2023-10-01-Preview View

## JSON query editor

```

1  {
2    "search": "*"
3  }

```

**Search**

## Results

```

1  {
2    "@odata.context": "https://free-demo-search-svc.search.windows.net/indexes('realestate-us-sample-index')/$met",
3    "@search.nextPageParameters": {
4      "search": "*",
5      "top": null,
6      "skip": 50
7    },
8    "value": [
9      {
10        "@search.score": 1,
11        "listingId": "OTM4Mzk1Mw2",
12        "beds": 5,
13        "baths": 4,
14        "description": "This is a flatlet and is move-in ready. Enjoy lake front property located close to parks",
15        "description_de": "Dies ist ein Allergiker und Einzug bereit. Genießen Sie See Immobilien Parks in der N",
16        "description_fr": "Il s'agit d'un appartement et est prêt à se déplacer en. Profitez de cette propriété",
17        "description_it": "Si tratta di una casetta e si muovono in pronto. Godere di proprietà fronte lago Situ",
18        "description_es": "Se trata de un flatlet y está lista para la mudanza. Disfrute de propiedad frente lag

```



## 자유 텍스트 검색

연산자가 있거나 없는 자유 형식 쿼리는 사용자 지정 앱에서 Azure AI 검색으로 보낸 사용자 정의 쿼리를 시뮬레이션하는 데 유용합니다. 인덱스 정의에서 "검색 가능"으로 특성이 지정된 필드만 일치 항목에 대해 검색됩니다.

무료 텍스트 쿼리에는 JSON 보기가 필요하지 않지만 이 문서에서는 다른 예제와의 일관성을 위해 JSON으로 제공합니다.

쿼리 용어나 식 같은 검색 조건을 입력하면 검색 순위가 매겨집니다. 다음 예제에서는 자유 텍스트 검색을 보여 줍니다. "@search.score"는 [기본 점수 알고리즘](#)을 사용하여 일치에 대해 계산된 관련성 점수입니다.

## JSON

```
{
  "search": "Seattle townhouse `Lake Washington` miele OR thermador
appliance"
}
```

## 결과

특정 관심 용어의 경우 Ctrl+F를 사용하여 결과 내에서 검색을 수행할 수 있습니다.

Results

Aa ab \*
1 of 42
↑ ↓ = ×

```

1 √ {
2   "@odata.context": "https://free-demo-search-svc.search.windows.net/indexes('realestate-us-sample-index')/$met
3   "@search.nextPageParameters": {
4     "search": "Seattle townhouse `Lake Washington` miele OR thermador appliance",
5     "top": null,
6     "skip": 50
7   },
8   "value": [
9     {
10       "@search.score": 47.812214,
11       "listingId": "OTM4MzMyQ2",
12       "beds": 3,
13       "baths": 1,
14       "description": "This is a townhouse and is well maintained. Enjoy lake front property located in a cul-de-sac. It has 3 bedrooms, 1 bath, and a large deck overlooking the lake. The kitchen features stainless steel appliances including a Thermador double oven and a Miele dishwasher. The living room has a fireplace and plenty of natural light. The master bedroom includes a walk-in closet and an en-suite bathroom. The property is located in a quiet neighborhood with easy access to local amenities. Don't miss out on this opportunity to own a beautiful home by the water!",
15       "description_de": "Dies ist ein Stadthaus und ist sehr gepflegt. Genießen Sie See Immobilien gelegen in einer Sackgasse. Es hat 3 Schlafzimmer, 1 Bad und eine große Terrasse über dem See. Die Küche verfügt über Edelstahlgeräte einschließlich eines Thermador Doppelherdes und einer Miele Spülmaschine. Das Wohnzimmer hat eine Kamin und viel natürliches Licht. Das Hauptgeschoss verfügt über einen Walk-in-Kleiderschrank und eine eigene Badezimmer. Das Anwesen liegt in einer ruhigen Umgebung mit leichtem Zugang zu lokalen Einrichtungen. Fassen Sie diese Gelegenheit nicht aus!",
16       "description_fr": "Il s'agit d'une maison de ville et est bien entretenue. Profitez de cette propriété fronte lago située dans une impasse. Elle dispose de 3 chambres à coucher, 1 salle de bain et une grande terrasse donnant sur le lac. La cuisine est équipée d'appareils en acier inoxydable, dont un four à deux fours Thermador et une lave-vaisselle Miele. La pièce à vivre comprend un poêle à bois et offre beaucoup de lumière naturelle. La chambre principale possède un placard vestimentaire et une salle de bain en suite. L'habitation se trouve dans un quartier paisible avec un accès facile aux commodités locales. Ne manquez pas cette chance d'acheter une belle maison au bord du lac !",
17       "description_it": "Si tratta di una casa a schiera ed è ben tenuta. Godere di proprietà fronte lago Situata in una strada senza uscite. Dispone di 3 camere da letto, 1 bagno e una grande terrazza che guarda sul lago. La cucina ha gli apparecchi in acciaio inox, compreso un forno a due fuochi Thermador e una lavastoviglie Miele. Il soggiorno ha un caminetto e molta luce naturale. La camera da letto principale ha un armadio e un bagno privato. La casa si trova in un quartiere tranquillo con un facile accesso alle comodità locali. Non perdere questa occasione per acquistare una bella casa al lago !",
18       "description_es": "Se trata de una casa y bien mantenida. Disfrute de propiedades frente lago situadas en una calle sin salida. Ofrece 3 dormitorios, 1 baño y una gran terraza que mira al lago. La cocina tiene electrodomésticos de acero inoxidable, incluyendo un horno a dos hornos Thermador y una lavavajillas Miele. El salón tiene un hogar y mucha luz natural. La habitación principal tiene un armario y un baño privado. La casa se encuentra en un vecindario tranquilo con fácil acceso a las comodidades locales. No pierda la oportunidad de adquirir una hermosa casa al lado del agua !",
19       "description_pl": "To miesiąc sie w kamienicy i jest dobrze utrzymany. Ciesz się lake front Własność położona jest w ulicy bez wyjścia. Ma 3 pokoje, 1 łazienkę i dużą tarasą, która patrzy na jezioro. Kuchnia posiada sprzęt zainless steel, w tym dwufunkcyjny piekarnik Thermador i myczarka Miele. Salon posiada kominkę i wiele światła naturalnego. Główna sypialnia ma szafę i łazienkę prywatną. Dom położony jest w spokojnym sąsiedztwie z łatwym dostępem do lokalnych instalacji. Nie przegap tej okazji, aby kupić pięknego domu nad wodą !",
20       "description_nl": "Dit is een herenhuis en is goed onderhouden. Geniet van lake eigendom gelegen in een straat zonder uitgang. Het heeft 3 slaapkamers, 1 badkamer en een grote terras die uitzicht heeft op het meer. De keuken is voorzien van edelstaalapparatuur, waaronder een dubbele oven Thermador en een wasmachine Miele. De woonkamer heeft een open haard en veel natuurlijk licht. De hoofdslaapkamer heeft een kledingkast en een eigen badkamer. De woning ligt in een rustige omgeving met gemakkelijke toegang tot lokale voorzieningen. Mis deze kans niet om een prachtige woning aan het water te kopen !",
21       "sqft": 1944,
22       "daysOnMarket": 48,
23       "status": "pending",
24       "source": "Watson Realty",
    }
  ]
}

```

## 일치하는 문서 수

인덱스에서 찾은 일치 항목 수를 가져오려면 `"count": true`(을)를 추가합니다. 빈 검색에서 count는 인덱스의 총 문서 수입니다. 정규화된 검색에서는 쿼리 입력과 일치하는 문서의 수입니다. 서비스는 기본적으로 상위 50개의 일치 항목을 반환하므로 결과에 반환된 항목보다 인덱스에서 더 많은 일치 항목이 표시될 수 있습니다.

JSON

```
{
  "search": "Seattle townhouse `Lake Washington` miele OR thermador
  appliance",
  "count": true
}
```

## 결과

JSON query editor

```

1  {
2    "search": "Seattle townhouse `Lake Washington` miele OR thermador appliance",
3    "count": true
4  }

```

Results

Search

```

1  {
2    "@odata.context": "https://free-demo-search-svc.search.windows.net/indexes('realestate-us-sample-index')/$metadata",
3    "@odata.count": 3710,
4    "@search.nextPageParameters": {
5      "search": "Seattle townhouse `Lake Washington` miele OR thermador appliance",
6      "count": true,
7      "top": null,
8      "skip": 50
9    },
10   "value": [
11     {
12       "@search.score": 47.812214,
13       "listingId": "OTM4MzMyQ2",
14       "beds": 3,
15       "baths": 1,
16       "description": "This is a townhouse and is well maintained. Enjoy lake front property located in a cul-de-sac. Large open floor plan with high ceilings, large windows overlooking the lake, and a spacious deck with a hot tub. The kitchen features stainless steel appliances, including a Miele oven and a Thermador range. The bedrooms are spacious and well-lit, with a large master suite featuring a walk-in closet and a ensuite bathroom. The finished basement includes a rec room, a full bathroom, and a laundry room. The property is located in a highly sought-after neighborhood with excellent schools and amenities. Don't miss this opportunity to own a piece of paradise!",
17       "description_de": "Dies ist ein Stadthaus und ist sehr gepflegt. Genießen Sie See Immobilien gelegen in einer Sackgasse. Große offene Raumplanung mit hohen Decken, großen Fenstern, die den See überblicken, und einer geräumigen Terrasse mit einer Whirlpool. Die Küche verfügt über Edelstahlgeräte, einschließlich einer Miele Ofen und einer Thermador Herd. Die Schlafzimmer sind groß und hell, mit einem großen Hauptbad mit einer Walk-in-Kleiderschrank und einer Ensuite-Badezimmer. Das fertiggestellte Untergeschoss umfasst eine Rec-Zimmer, ein vollständiges Badezimmer und eine Waschküche. Das Objekt liegt in einer hoch gesuchten Nachbarschaft mit exzellenten Schulen und Einrichtungen. Fassen Sie diese Gelegenheit nicht aus!",
18       "description_fr": "Il s'agit d'une maison de ville et est bien entretenue. Profitez de cette propriété située dans une impasse. Une grande planification ouverte avec des plafonds élevés, de grands fenêtres qui donnent sur le lac, et une terrasse spacieuse avec un spa. La cuisine offre des appareils en acier inoxydable, dont un four Miele et une cuisinière Thermador. Les chambres sont spacieuses et bien éclairées, avec une grande chambre principale dotée d'un placard à accès direct et d'une salle de bain en suite. Le sous-sol terminé comprend une pièce de récréation, un bain complet et une laverie. L'endroit se trouve dans une communauté très recherchée avec des écoles et des installations exceptionnelles. Ne manquez pas cette chance de posséder un morceau de paradis !",
19       "description_it": "Si tratta di una casa a schiera ed è ben tenuta. Godere di questa proprietà situata in una strada a fondo perduto. Grande pianta aperta con soffitti alti, grandi finestre che guardano sul lago, e una terrazza ampia con un whirlpool. La cucina ha apparecchiature in acciaio inox, compreso un forno Miele e una cucina Thermador. I camere sono spaziose e ben illuminate, con una grande camera da letto principale con un armadio a accesso diretto e un bagno da bagno. Il seminterrato finito comprende una stanza di ricreazione, un bagno completo e una lavandaia. Il luogo si trova in una comunità molto richiesta con scuole e installazioni eccezionali. Non perdere questa occasione di avere un pezzo di paradiso !",
20       "description_es": "Se trata de una casa a una sola planta y está bien mantenida. Disfrute de esta propiedad ubicada en una calle sin salida. Una amplia planta abierta con techos altos, grandes ventanas que ofrecen una vista al lago, y una terraza amplia con un spa. La cocina tiene electrodomésticos de acero inoxidable, incluyendo un horno Miele y una cocina Thermador. Las habitaciones son espaciosas y bien iluminadas, con una gran habitación principal que incluye un armario empotrado y un baño en suite. El sótano terminado incluye una sala de estar, un baño completo y una lavadora. El lugar se encuentra en una comunidad muy demandada con escuelas y instalaciones excepcionales. No pierda la oportunidad de tener un pedazo de paraíso !",
21       "description_pl": "To miesci się w kamienicy i jest dobrze utrzymany. Ciesz się lake front Własciwosc położonej w ślepym uliczce. Duża otwarta przestrzeń z wysokimi sufitami, dużymi oknami, które dają widok na jezioro, oraz dużą tarasą z jacuzzi. Kuchnia posiada sprzęt szklano-żelazowy, w tym piekarnik Miele i piekarnik Thermador. Sypialnie są duże i dobrze oświetlone, z dużą sypialnią z garderobą i łazienką z prysznicem. Poddasze skończone zawiera salon, łazienkę pełną i pralnię. Miejsce to znajduje się w bardzo popularnej dzielnicy z wyjątkowymi szkołami i infrastrukturą. Nie przegap tej okazji, aby mieć kawałek nieba !",
22       "description_nl": "Dit is een herenhuis en is goed onderhouden. Geniet van lake front eigendom gelegen in een doodlopende straat. Groot open concept met hoge plafonds, grote ramen die u uitzicht biedt op het meer, en een grote terras met een whirlpool. De keuken heeft edele stalen apparatuur, waaronder een oven Miele en een Thermador kookplaat. De slaapkamers zijn ruim en goed verlicht, met een grote hoofdslaapkamer die een groot garderobekastje en een badkamer met een douche heeft. Het souterrain is volledig ingericht met een woonkamer, een volledige badkamer en een wasruimte. Het huis ligt in een zeer gewilde wijk met uitmuntende scholen en voorzieningen. Wees erop dat je deze kans niet mist om een stukje hemel te hebben !",
23       "sqft": 1944,
24       "daysOnMarket": 48,

```

## 검색 결과의 필드 제한

검색 탐색기에서 더 읽기 쉬운 출력을 위해 명시적으로 이름이 지정된 필드로 결과를 제한하려면 `"select"`를 추가합니다. 검색 인덱스에서 "검색 가능"으로 표시된 필드만 결과에 표시될 수 있습니다.

JSON

```
{
  "search": "seattle condo",
  "count": true,
  "select": "listingId, beds, baths, description, street, city, price"
}
```

## 결과

JSON query editor

```

1  {
2    "search": "seattle condo",
3    "count": true,
4    "select": "listingId, beds, baths, description, street, city, price"
5  }

```

Results

Search

```

1  {
2    "@odata.context": "https://free-demo-search-svc.search.windows.net/indexes('realestate-us-sample-index')/$met...
3    "@odata.count": 2056,
4    "@search.nextPageParameters": {
5      "search": "seattle condo",
6      "select": "listingId, beds, baths, description, street, city, price",
7      "count": true,
8      "top": null,
9      "skip": 50
10 },
11 "value": [
12   {
13     "@search.score": 13.861284,
14     "listingId": "OTM4Mjg2Ng2",
15     "beds": 2,
16     "baths": 2,
17     "description": "This is a condo and is a beautiful home. Enjoy oceanfrontage located close to a river and a ...
18     "street": "24th Avenue South",
19     "city": "Seattle",
20     "price": 637632
21   },
22   {
23     "@search.score": 13.215699,
24     "listingId": "OTM4MjMyNw2",
25     "beds": 4,
26     "baths": 4,
27     "description": "This is a condo and is freshly painted. Enjoy oceanfrontage located in a cultisac and fe...
28     "street": "32nd Avenue West",
29     "city": "Seattle",

```

## 결과의 다음 일괄 처리 반환

Azure AI 검색은 검색 순위에 따라 상위 50개 일치 항목을 반환합니다. 일치하는 다음 문서 집합을 가져오려면 `"top": 100` 및 `"skip": 50`(을)를 추가하여 결과 집합을 100개 문서 (기본값은 50개, 최댓값은 1000개)로 늘리고 처음 50개 문서는 건너뜁니다. 문서 키 (`listingID`)를 확인하여 문서를 식별할 수 있습니다.

순위가 지정된 결과를 얻으려면 쿼리 용어 또는 식과 같은 검색 조건을 제공해야 합니다. 검색 결과에 더 깊게 도달할수록 검색 점수가 줄어듭니다.

JSON

```
{
  "search": "seattle condo",
  "count": true,
  "select": "listingId, beds, baths, description, street, city, price",
  "top": 100,
  "skip": 50
}
```

## 결과

JSON query editor

```

1 √ {
2   "search": "seattle condo",
3   "count": true,
4   "select": "listingId, beds, baths, description, street, city, price",
5   "top": 100,
6   "skip": 50
7 }

```

Results

Search

```

1 {
2   "@odata.context": "https://free-demo-search-svc.search.windows.net/indexes('realestate-us-sample-index')/$met
3   "@odata.count": 2056,
4   "value": [
5     {
6       "@search.score": 12.719713,
7       "listingId": "OTM4MzQ1Mg2",
8       "beds": 4,
9       "baths": 4,
10      "description": "This is a condo and is a dream home. This property has mountain views located in a cul-de-s
11      "street": "33rd Avenue",
12      "city": "Seattle",
13      "price": 2260224
14    },
15    {
16      "@search.score": 12.719713,
17      "listingId": "OTM4Mjky0Q2",
18      "beds": 4,
19      "baths": 2,
20      "description": "This is a condo and is priced to sell. This home provides coastal views located close to the
21      "street": "Ellis Avenue South",
22      "city": "Seattle",
23      "price": 1202688
24    },

```

## 필터 식(보다 큼, 보다 작음, 같음)

`filter` 매개 변수를 사용하여 포함 또는 제외 조건을 지정합니다. 필드는 인덱스에서 "필터링 가능"으로 특성이 지정되어야 합니다. 이 예제에서는 침실 개수가 4개 이상인 항목을 검색합니다.

JSON

```
{
  "search": "seattle condo",
  "count": true,
  "select": "listingId, beds, baths, description",
  "filter": "beds gt 3"
}
```

## 결과

JSON query editor

```

1  {
2    "search": "seattle condo",
3    "count": true,
4    "select": "listingId, beds, baths, description",
5    "filter": "beds gt 3"
6  }

```

Search

Results

```

12  "value": [
13    {
14      "@search.score": 13.215699,
15      "listingId": "OTM4MjMyNw2",
16      "beds": 4,
17      "baths": 4,
18      "description": "This is a condo and is freshly painted. Enjoy oceanfrontage located in a cultisac and fe."
19    },
20    {
21      "@search.score": 13.215699,
22      "listingId": "OTM4MzI3MQ2",
23      "beds": 4,
24      "baths": 4,
25      "description": "This is a condo and is well maintained. This property has lake access located in a culti."
26    },
27    {
28      "@search.score": 12.968667,
29      "listingId": "OTM4MjkzNA2",
30      "beds": 5,
31      "baths": 2,
32      "description": "This is a condo and is priced to sell. Enjoy lake front property located close to school"
33    },

```



## 결과 정렬

검색 점수 외에 다른 필드를 기준으로 결과를 정렬하려면 `orderby(을)`를 추가합니다. 필드는 인덱스에서 "정렬 가능"으로 특성이 지정되어야 합니다. 필터링된 값이 동일한 경우 (예: 동일한 가격) 순서는 임의적이지만 더 심층적인 정렬을 위해 더 많은 조건을 추가할 수 있습니다. 다음은 이 출력을 테스트하는 데 사용할 수 있는 예제 식입니다.

JSON

```
{
  "search": "seattle condo",
  "count": true,
  "select": "listingId, price, beds, baths, description",
  "filter": "beds gt 3",
  "orderby": "price asc"
}
```

## 결과

JSON query editor

```

1  {
2    "search": "seattle condo",
3    "count": true,
4    "select": "listingId, price, beds, baths, description",
5    "filter": "beds gt 3",
6    "orderby": "price asc"
7  }

```

Search

Results

```

13  "value": [
14    {
15      "@search.score": 1.0539877,
16      "listingId": "OTM4NTIxNA2",
17      "beds": 4,
18      "baths": 1,
19      "description": "This is a two-story residence and is brand new. This home provides panoramic views located in the heart of Seattle's Lake Washington area. It features 4 bedrooms, 1 bathroom, and an open floor plan with high ceilings and large windows. The kitchen is equipped with stainless steel appliances and granite countertops. The master bedroom includes an en-suite bathroom. The property is surrounded by lush landscaping and has a private deck with outdoor seating. It is perfect for entertaining or everyday living. Call today to schedule a viewing!",
20      "price": 518400
21    },
22    {
23      "@search.score": 1.0539877,
24      "listingId": "OTM4MjEyNg2",
25      "beds": 4,
26      "baths": 1,
27      "description": "This is a two-story residence and is a short sale. Enjoy gleaming hardwood floors located in the heart of Seattle's Lake Washington area. It features 4 bedrooms, 1 bathroom, and an open floor plan with high ceilings and large windows. The kitchen is equipped with stainless steel appliances and granite countertops. The master bedroom includes an en-suite bathroom. The property is surrounded by lush landscaping and has a private deck with outdoor seating. It is perfect for entertaining or everyday living. Call today to schedule a viewing!",
28      "price": 518400
29    },
30    {
31      "@search.score": 1.0539877,
32      "listingId": "OTM4MjkyMA2",
33      "beds": 4,
34      "baths": 1,
35      "description": "This is a two-story residence and is a beautiful home. This property has great views located in the heart of Seattle's Lake Washington area. It features 4 bedrooms, 1 bathroom, and an open floor plan with high ceilings and large windows. The kitchen is equipped with stainless steel appliances and granite countertops. The master bedroom includes an en-suite bathroom. The property is surrounded by lush landscaping and has a private deck with outdoor seating. It is perfect for entertaining or everyday living. Call today to schedule a viewing!",
36      "price": 520992
37    },

```

## 핵심 내용

이 빠른 시작에서는 검색 탐색기에서 REST API를 사용하여 인덱스를 쿼리했습니다.

- 결과는 자세한 JSON 문서로 반환되므로 문서 생성 및 콘텐츠를 완전히 볼 수 있습니다. 쿼리 식의 `select` 매개 변수는 반환되는 필드를 제한할 수 있습니다.
- 검색 결과는 인덱스에서 "검색 가능"으로 표시된 모든 필드로 구성됩니다. 특성을 검토하려면 인접한 **필드** 탭을 선택합니다.
- 상용 웹 브라우저에 입력하는 것과 유사한 키워드 검색은 최종 사용자 환경을 테스트하는 데 유용합니다. 예를 들어, 기본 제공 부동산 샘플 인덱스를 가정할 경우 "Seattle apartments lake washington"을 입력한 후 Ctrl+F를 사용하여 검색 결과 내에서 용어를 찾을 수 있습니다.
- 쿼리 및 필터 식은 Azure AI 검색에서 구현되는 구문으로 표현됩니다. 기본 구문은 [간단한 구문](#)이지만 필요에 따라 보다 강력한 쿼리를 위해 [전체 Lucene](#)을 사용할 수 있습니다. [필터 식](#)은 OData 구문으로 표현됩니다.

## 리소스 정리

자체 구독으로 작업하는 경우 프로젝트가 끝날 때 만든 리소스가 여전히 필요한지 여부를 결정하는 것이 좋습니다. 계속 실행되는 리소스에는 요금이 부과될 수 있습니다. 리소

스를 개별적으로 삭제하거나 리소스 그룹을 삭제하여 전체 리소스 세트를 삭제할 수 있습니다.

왼쪽 탐색 창의 **모든 리소스** 또는 **리소스 그룹** 링크를 사용하여 포털에서 리소스를 찾고 관리할 수 있습니다.

무료 서비스를 사용하는 경우 인덱스, 인덱서, 데이터 원본 3개로 제한됩니다. 포털에서 개별 항목을 삭제하여 제한 이하로 유지할 수 있습니다.

## 다음 단계

쿼리 구조 및 구문에 대해 자세히 알아보려면 REST 클라이언트를 사용하여 API의 더 많은 파트를 사용하는 쿼리 식을 만드세요. [검색 POST REST API](#)는 학습 및 탐색에 특히 유용합니다.

[REST에서 기본 쿼리 만들기](#)

---

## 피드백

이 페이지가 도움이 되었나요?

 Yes

 No

[제품 사용자 의견 제공](#) | [Microsoft Q&A에서 도움말 보기](#)

# 빠른 시작: REST를 사용한 키워드 검색

아티클 • 2024. 10. 31.

Azure AI 검색의 REST API는 미리 보기 기능을 포함하여 모든 기능에 프로그래밍 방식으로 액세스할 수 있도록 하며 기능이 작동하는 방식을 쉽게 알아볼 수 있습니다. 이 빠른 시작에서 [검색 REST API](#)를 호출하여 Azure AI 검색에서 인덱스를 만들고 로드하고 쿼리하는 방법을 알아봅니다.

Azure 구독이 아직 없는 경우 시작하기 전에 [체험 계정](#)을 만듭니다.

## 필수 조건

- [Visual Studio Code](#) 와 [REST 클라이언트](#).
- [Azure AI 검색](#). 현재 구독에서 기존 Azure AI 검색 리소스를 만들거나 찾습니다. 이 빠른 시작에서는 체험 서비스를 사용할 수 있습니다.

## 파일 다운로드

GitHub에서 [REST 샘플을 다운로드](#)하여 이 빠른 시작에서 요청을 보냅니다. 지침은 [GitHub에서 파일 다운로드](#)에서 확인할 수 있습니다.

이 문서의 지침을 사용하여 로컬 시스템에서 새 파일을 시작하고 수동으로 요청을 만들 수도 있습니다.

## 검색 서비스 엔드포인트 가져오기

Azure Portal에서 검색 서비스 엔드포인트를 찾을 수 있습니다.

1. [Azure Portal](#)에 로그인하고 [검색 서비스를 찾습니다](#).
2. 개요 홈페이지에서 URL을 찾습니다. 엔드포인트의 예는 다음과 같습니다.

`https://mydemo.search.windows.net`

The screenshot shows the Azure Portal interface for a search service named "free-search-demo-svc". The "Overview" tab is selected. In the top navigation bar, there are links for "Search", "Add index", "Import data", "Import and vectorize data", "Search explorer", "Refresh", "Delete", and "Move". Below the navigation, there's a "View Cost" and "JSON View" button. The main content area has sections for "Essentials" and "Activity log". The "Essentials" section displays resource group, location (East US), subscription, and status information. A table provides details for the "Url": https://free-search-demo-svc.search.windows.net. The "Activity log" section is also visible on the left.

이후 단계에서 이 엔드포인트를 `.rest` 또는 `.http` 파일에 붙여넣습니다.

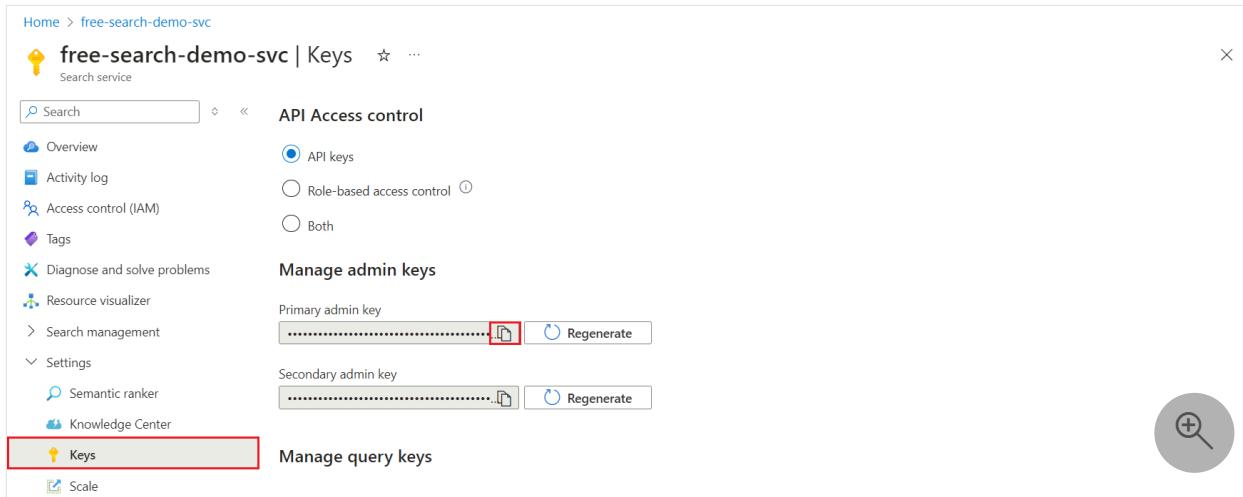
## 액세스 구성

검색 엔드포인트에 대한 요청은 인증 및 권한 부여를 받아야 합니다. 이 작업에 API 키 또는 역할을 사용할 수 있습니다. 키는 시작하기가 더 쉽지만 역할이 더 안전합니다.

역할 기반 연결의 경우 다음 지침에 따라 클라이언트 앱의 ID가 아닌 사용자 ID로 Azure AI 검색에 연결합니다.

### 옵션 1: 키 사용

[설정>키를 선택한 다음, 관리자 키를 복사합니다. 관리자 키는 개체를 추가, 수정, 삭제하는 데 사용됩니다. 교환 가능한 관리자 키는 2개입니다. 둘 중 하나를 복사합니다. 자세한 내용은 \[키 인증을 사용하여 Azure AI 검색에 연결\]\(#\)을 참조하세요.](#)



이후 단계에서 이 키를 `.rest` 또는 `.http` 파일에 붙여넣습니다.

### 옵션 2: 역할 사용

검색 서비스가 [역할 기반 액세스](#)에 대해 구성되어 있는지 확인합니다. [개발자 액세스를 위한 역할 할당](#)이 미리 구성되어 있어야 합니다. 역할 할당은 검색 인덱스를 생성, 로드 및 쿼리할 수 있는 권한을 부여해야 합니다.

이 섹션에서는 Azure CLI, Azure PowerShell 또는 Azure Portal을 사용하여 개인 ID 토큰을 가져옵니다.

#### Azure CLI

1. Azure CLI에 로그인합니다.

```
Azure CLI
```

```
az login
```

2. 개인 ID 토큰을 가져옵니다.

```
Azure CLI
```

```
az account get-access-token --scope  
https://search.azure.com/.default
```

이후 단계에서 개인 ID 토큰을 `.rest` 또는 `.http` 파일에 붙여넣습니다.

### ① 참고

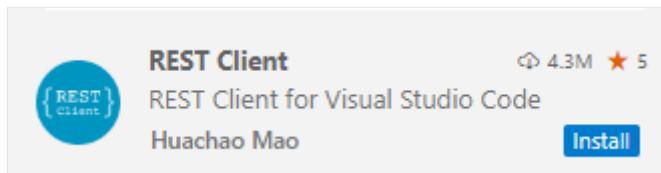
이 섹션에서는 사용자를 대신하여 Azure AI 검색에 연결하는 로컬 클라이언트를 사용한다고 가정합니다. 애플리케이션이 Microsoft Entra ID에 등록되어 있다고 가정하고 클라이언트 앱에 대한 토큰을 가져오는 방식을 사용할 수도 있습니다.

## Visual Studio Code 설정

Visual Studio Code용 REST 클라이언트에 익숙하지 않은 경우 이 섹션에는 이 빠른 시작의 작업을 완료할 수 있도록 설정이 포함되어 있습니다.

1. Visual Studio Code를 시작하고 **확장** 탭을 선택합니다.

2. REST 클라이언트를 검색하고 설치를 선택합니다.



3. `.rest` 또는 `.http` 파일 확장자를 사용하여 명명된 새 파일을 열거나 만듭니다.

4. API 키를 사용하는 경우 다음 예를 붙여넣습니다. `@baseUrl` 및 `@apiKey` 자리 표시자를 이전에 복사한 값으로 바꿉니다.

```
HTTP
```

```
@baseUrl = PUT-YOUR-SEARCH-SERVICE-ENDPOINT-HERE  
@apiKey = PUT-YOUR-SEARCH-SERVICE-API-KEY-HERE
```

```

### List existing indexes by name
GET {{baseUrl}}/indexes?api-version=2024-07-01&$select=name HTTP/1.1
Content-Type: application/json
api-key: {{apiKey}}

```

5. 또는 역할을 사용하는 경우 이 예를 붙여넣습니다. @baseUrl 및 @token 자리 표시자를 이전에 복사한 값으로 바꿉니다.

```

HTTP

@baseUrl = PUT-YOUR-SEARCH-SERVICE-ENDPOINT-HERE
@token = PUT-YOUR-PERSONAL-IDENTITY-TOKEN-HERE

### List existing indexes by name
GET {{baseUrl}}/indexes?api-version=2024-07-01&$select=name HTTP/1.1
Content-Type: application/json
Authorization: Bearer {{token}}

```

6. 요청 보내기를 선택합니다. 인접한 창에 응답이 표시됩니다. 기존 인덱스가 있는 경우 인덱스가 나열됩니다. 그러지 않으면 목록은 비어 있습니다. HTTP 코드가 200 OK 인 경우 다음 단계를 수행할 준비가 된 것입니다.

```

HTTP/1.1 200 OK
Transfer-Encoding: chunked
Content-Type: application/json; odata.metadata=minimal; odata.streaming=true; charset=utf-8
Content-Encoding: gzip
Vary: Accept-Encoding
Server: Microsoft-IIS/10.0
Strict-Transport-Security: max-age=2592000, max-age=15724800; includeSubDomains
Preference-Applied: odata.include-annotations="*"
OData-Version: 4.0
request-id: 24a5a81a-2b1d-4f42-abdd-bef9275531fb
elapsed-time: 184
Date: Wed, 06 Mar 2024 17:31:38 GMT
Connection: close
15 ↴ {
16
17 ↴   "value": [
18 ↴     {
19 ↴       "name": "good-books"
20 ↴     },
21 ↴     {
22 ↴       "name": "demoindex"
23 ↴     },

```

주요 정보:

- 매개 변수는 @ 접두사를 사용하여 지정됩니다.
- ### 은 REST 호출을 지정합니다. 다음 줄에는 HTTP/1.1 을 포함해야 하는 요청이 포함됩니다.
- Send request 가 요청 위에 표시되어야 합니다.

## 인덱스 만들기

.rest 파일에 두 번째 요청을 추가합니다. 인덱스 만들기(REST)는 검색 인덱스를 만들고 검색 서비스에 물리적 데이터 구조를 설정합니다.

# 1. 다음 예제를 붙여넣어 검색 서비스에 `hotels-quickstart` 인덱스를 만듭니다.

HTTP

```
### Create a new index
POST {{baseUrl}}/indexes?api-version=2024-07-01 HTTP/1.1
Content-Type: application/json
Authorization: Bearer {{token}}


{
    "name": "hotels-quickstart",
    "fields": [
        {"name": "HotelId", "type": "Edm.String", "key": true,
 "filterable": true},
        {"name": "HotelName", "type": "Edm.String", "searchable": true,
 "filterable": false, "sortable": true, "facetable": false},
        {"name": "Description", "type": "Edm.String", "searchable": true,
 "filterable": false, "sortable": false, "facetable": false,
 "analyzer": "en.lucene"},
        {"name": "Category", "type": "Edm.String", "searchable": true,
 "filterable": true, "sortable": true, "facetable": true},
        {"name": "Tags", "type": "Collection(Edm.String)",
 "searchable": true, "filterable": true, "sortable": false, "facetable": true},
        {"name": "ParkingIncluded", "type": "Edm.Boolean",
 "filterable": true, "sortable": true, "facetable": true},
        {"name": "LastRenovationDate", "type": "Edm.DateTimeOffset",
 "filterable": true, "sortable": true,
 "facetable": true},
        {"name": "Rating", "type": "Edm.Double", "filterable": true,
 "sortable": true, "facetable": true},
        {"name": "Address", "type": "Edm.ComplexType",
 "fields": [
            {"name": "StreetAddress", "type": "Edm.String",
 "filterable": false, "sortable": false, "facetable": false,
 "searchable": true},
            {"name": "City", "type": "Edm.String", "searchable": true,
 "filterable": true, "sortable": true, "facetable": true},
            {"name": "StateProvince", "type": "Edm.String",
 "searchable": true, "filterable": true, "sortable": true,
 "facetable": true},
            {"name": "PostalCode", "type": "Edm.String",
 "searchable": true, "filterable": true, "sortable": true,
 "facetable": true},
            {"name": "Country", "type": "Edm.String", "searchable": true,
 "filterable": true, "sortable": true, "facetable": true}
        ]
    }
}
```

2. 요청 보내기를 선택합니다. `HTTP/1.1 201 Created` 응답이 있어야 하며 응답 본문에 인덱스 스키마의 JSON 표현이 포함되어야 합니다.

Header name must be a valid HTTP token ["{}"] 오류가 발생하면 `api-key` 와 요청 본문 사이에 빈 줄이 있는지 확인합니다. HTTP 504가 표시될 경우 URL이 HTTPS를 지정하는지 확인합니다. HTTP 400 또는 404가 표시되는 경우 요청 본문에서 복사/붙여 넣기 오류가 없는지 확인합니다. HTTP 403은 일반적으로 API 키에 문제가 있음을 나타냅니다. 키가 잘못되었거나 API 키를 지정하는 방법에 문제가 있는 경우입니다.

이제 파일에 요청이 여러 개 있습니다. `###` 이 새 요청을 시작하고 각 요청은 독립적으로 실행됩니다.

```
C: > Users > OneDrive - Microsoft > Desktop > az-search-quickstart.rest > POST /indexes/hotels-quickstart/docs/index?api-version=2023-11-01
  3 references
1   @baseUrl = https://PUT-YOUR-SEARCH-SERVICE-NAME-HERE.search.windows.net
  3 references
2   @apiKey = 00000000000000000000000000000000
  3
4   ### List existing indexes by name
  Send Request
5   > GET {{baseUrl}}/indexes?api-version=2023-11-01&$select=name HTTP/1.1 ...
  9   ### Create a new index
  Send Request
10  > POST {{baseUrl}}/indexes?api-version=2023-11-01 HTTP/1.1 ...
  37   ### Upload documents
  Send Request
38  > POST {{baseUrl}}/indexes/hotels-quickstart/docs/index?api-version=2023-11-01 HTTP/1.1 ...
```

## 인덱스 정의 정보

인덱스 스키마 내에서 필드 컬렉션은 문서 구조를 정의합니다. 업로드하는 각 문서에는 이러한 필드가 있어야 합니다. 각 필드는 [EDM\(엔터티 데이터 모델\) 데이터 형식](#)에 할당되어야 합니다. 문자열 필드는 전체 텍스트 검색에 사용됩니다. 숫자 데이터를 검색 가능하게 하려면 데이터 형식이 `Edm.String` 인지 확인합니다. `Edm.Int32` 와 같은 다른 데이터 형식은 필터링, 정렬, 패싯 및 검색이 가능하지만 전체 텍스트 검색은 가능하지 않습니다.

필드의 특성에 따라 허용되는 작업이 결정됩니다. REST API는 [기본적으로 많은 작업을 허용합니다](#). 예를 들어, 기본적으로 모든 문자열을 검색할 수 있습니다. REST API의 경우 동작을 해제해야 하는 경우에만 특성이 필요할 수 있습니다.

JSON

```
{
  "name": "hotels-quickstart",
  "fields": [
    {"name": "HotelId", "type": "Edm.String", "key": true, "filterable": true},
    {"name": "HotelName", "type": "Edm.String", "searchable": true, "filterable": false, "sortable": true, "facetable": false},
    {"name": "Description", "type": "Edm.String", "searchable": true, "filterable": false, "sortable": false, "facetable": false, "analyzer": "standard"}
```

```

    "en.lucene"},

        {"name": "Category", "type": "Edm.String", "searchable": true,
"filterable": true, "sortable": true, "facetable": true},
            {"name": "Tags", "type": "Collection(Edm.String)", "searchable": true,
"filterable": true, "sortable": false, "facetable": true},
                {"name": "ParkingIncluded", "type": "Edm.Boolean", "filterable": true,
"sortable": true, "facetable": true},
                    {"name": "LastRenovationDate", "type": "Edm.DateTimeOffset",
"filterable": true, "sortable": true, "facetable": true},
                        {"name": "Rating", "type": "Edm.Double", "filterable": true,
"sortable": true, "facetable": true},
                            {"name": "Address", "type": "Edm.ComplexType",
"fields": [
                                {"name": "StreetAddress", "type": "Edm.String", "filterable": false,
"sortable": false, "facetable": false, "searchable": true},
                                {"name": "City", "type": "Edm.String", "searchable": true,
"filterable": true, "sortable": true, "facetable": true},
                                {"name": "StateProvince", "type": "Edm.String", "searchable": true,
"filterable": true, "sortable": true, "facetable": true},
                                {"name": "PostalCode", "type": "Edm.String", "searchable": true,
"filterable": true, "sortable": true, "facetable": true},
                                {"name": "Country", "type": "Edm.String", "searchable": true,
"filterable": true, "sortable": true, "facetable": true}
                            ]
                        }
                    }
                ]
            }
        }
    }
}

```

## 문서 로드

인덱스 만들기 및 로드는 별도의 단계입니다. Azure AI 검색에서 인덱스에는 검색 가능한 모든 데이터가 포함되어 있으며 검색 서비스에서 실행되는 쿼리가 있습니다. REST 호출의 경우 데이터는 JSON 문서로 제공됩니다. 이 작업에는 [문서- 인덱스 REST API](#)를 사용합니다.

`docs` 컬렉션 및 `index` 작업을 포함하도록 URI가 확장됩니다.

1. 다음 예제를 붙여넣어 JSON 문서를 검색 인덱스로 업로드합니다.

HTTP

```

### Upload documents
POST {{baseUrl}}/indexes/hotels-quickstart/docs/index?api-version=2024-07-01 HTTP/1.1
Content-Type: application/json
Authorization: Bearer {{token}}


{
    "value": [
        {

```

```
        "@search.action": "upload",
        "HotelId": "1",
        "HotelName": "Stay-Kay City Hotel",
        "Description": "The hotel is ideally located on the main commercial artery of the city in the heart of New York. A few minutes away is Time's Square and the historic centre of the city, as well as other places of interest that make New York one of America's most attractive and cosmopolitan cities.",
        "Category": "Boutique",
        "Tags": [ "pool", "air conditioning", "concierge" ],
        "ParkingIncluded": false,
        "LastRenovationDate": "1970-01-18T00:00:00Z",
        "Rating": 3.60,
        "Address":
        {
            "StreetAddress": "677 5th Ave",
            "City": "New York",
            "StateProvince": "NY",
            "PostalCode": "10022",
            "Country": "USA"
        }
    },
    {
        "@search.action": "upload",
        "HotelId": "2",
        "HotelName": "Old Century Hotel",
        "Description": "The hotel is situated in a nineteenth century plaza, which has been expanded and renovated to the highest architectural standards to create a modern, functional and first-class hotel in which art and unique historical elements coexist with the most modern comforts.",
        "Category": "Boutique",
        "Tags": [ "pool", "free wifi", "concierge" ],
        "ParkingIncluded": false,
        "LastRenovationDate": "1979-02-18T00:00:00Z",
        "Rating": 3.60,
        "Address":
        {
            "StreetAddress": "140 University Town Center Dr",
            "City": "Sarasota",
            "StateProvince": "FL",
            "PostalCode": "34243",
            "Country": "USA"
        }
    },
    {
        "@search.action": "upload",
        "HotelId": "3",
        "HotelName": "Gastronomic Landscape Hotel",
        "Description": "The Hotel stands out for its gastronomic excellence under the management of William Dough, who advises on and oversees all of the Hotel's restaurant services.",
        "Category": "Resort and Spa",
        "Tags": [ "air conditioning", "bar", "continental breakfast" ],
        "ParkingIncluded": true,
    }
```

```

    "LastRenovationDate": "2015-09-20T00:00:00Z",
    "Rating": 4.80,
    "Address":
    {
        "StreetAddress": "3393 Peachtree Rd",
        "City": "Atlanta",
        "StateProvince": "GA",
        "PostalCode": "30326",
        "Country": "USA"
    }
},
{
    "@search.action": "upload",
    "HotelId": "4",
    "HotelName": "Sublime Palace Hotel",
    "Description": "Sublime Palace Hotel is located in the heart of the historic center of Sublime in an extremely vibrant and lively area within short walking distance to the sites and landmarks of the city and is surrounded by the extraordinary beauty of churches, buildings, shops and monuments. Sublime Palace is part of a lovingly restored 1800 palace.",
    "Category": "Boutique",
    "Tags": [ "concierge", "view", "24-hour front desk service" ],
    "ParkingIncluded": true,
    "LastRenovationDate": "1960-02-06T00:00:00Z",
    "Rating": 4.60,
    "Address":
    {
        "StreetAddress": "7400 San Pedro Ave",
        "City": "San Antonio",
        "StateProvince": "TX",
        "PostalCode": "78216",
        "Country": "USA"
    }
}
]
}

```

2. 요청 보내기를 선택합니다. 몇 초 후에 인접한 창에 HTTP 201 응답이 표시됩니다.

207이 표시될 경우 하나 이상의 문서를 업로드하지 못했습니다. 404가 표시될 경우 요청의 헤더 또는 본문에 구문 오류가 있습니다. `/docs/index`를 포함하도록 엔드포인트를 변경했는지 확인합니다.

## 쿼리 실행

이제 문서가 로드되었으므로 [문서 - 게시물 검색\(REST\)](#)을 사용하여 문서에 대한 쿼리를 실행할 수 있습니다.

`/docs/search` 연산자를 사용하여 지정된 쿼리 식을 포함하도록 URI가 확장됩니다.

1. 다음 예제를 붙여넣어 검색 인덱스를 쿼리합니다. 그런 다음, **요청 보내기**를 선택합니다. 텍스트 검색 요청에는 항상 `search` 매개 변수가 포함됩니다. 이 예제에는 텍스트 검색을 특정 필드로 제한하는 선택적 `searchFields` 매개 변수가 포함되어 있습니다.

HTTP

```
### Run a query
POST {{baseUrl}}/indexes/hotels-quickstart/docs/search?api-
version=2024-07-01 HTTP/1.1
Content-Type: application/json
Authorization: Bearer {{token}}


{
  "search": "lake view",
  "select": "HotelId, HotelName, Tags, Description",
  "searchFields": "Description, Tags",
  "count": true
}
```

2. 인접한 창에서 응답을 검토합니다. 인덱스에서 찾은 일치 항목 수를 나타내는 개수, 관련성을 나타내는 검색 점수 및 `select` 문에 나열된 각 필드의 값이 있어야 합니다.

JSON

```
{
  "@odata.context": "https://my-
demo.search.windows.net/indexes('hotels-
quickstart')/$metadata#docs(*)",
  "@odata.count": 1,
  "value": [
    {
      "@search.score": 0.6189728,
      "HotelId": "4",
      "HotelName": "Sublime Palace Hotel",
      "Description": "Sublime Palace Hotel is located in the heart of
the historic center of Sublime in an extremely vibrant and lively area
within short walking distance to the sites and landmarks of the city
and is surrounded by the extraordinary beauty of churches, buildings,
shops and monuments. Sublime Palace is part of a lovingly restored 1800
palace.",
      "Tags": [
        "concierge",
        "view",
        "24-hour front desk service"
      ]
    }
  ]
}
```

```
]  
}
```

## 인덱스 속성 가져오기

[통계 가져오기](#)를 사용하여 문서 개수와 인덱스 크기를 쿼리할 수도 있습니다.

- 다음 예제를 붙여넣어 검색 인덱스를 쿼리합니다. 그런 다음, [요청 보내기](#)를 선택합니다.

HTTP

```
### Get index statistics  
GET {{baseUrl}}/indexes/hotels-quickstart/stats?api-version=2024-07-01  
HTTP/1.1  
Content-Type: application/json  
Authorization: Bearer {{token}}
```

- 응답을 검토합니다. 이 작업은 인덱스 스토리지에 대한 세부 정보를 쉽게 가져올 수 있는 방법입니다.

JSON

```
{  
    "@odata.context": "https://my-  
demo.search.windows.net/$metadata#Microsoft.Azure.Search.V2023_11_01.In  
dexStatistics",  
    "documentCount": 4,  
    "storageSize": 34707,  
    "vectorIndexSize": 0  
}
```

## 리소스 정리

본인 소유의 구독으로 이 모듈을 진행하고 있는 경우에는 프로젝트가 끝날 때 여기에서 만든 리소스가 계속 필요한지 확인하는 것이 좋습니다. 계속 실행되는 리소스에는 요금이 부과될 수 있습니다. 리소스를 개별적으로 삭제하거나 리소스 그룹을 삭제하여 전체 리소스 세트를 삭제할 수 있습니다.

가장 왼쪽 창의 **모든 리소스** 또는 **리소스 그룹** 링크를 사용하여 포털에서 리소스를 찾고 관리할 수 있습니다.

이 **DELETE** 명령을 사용해 볼 수도 있습니다.

## HTTP

```
### Delete an index
DELETE {{baseUrl}}/indexes/hotels-quickstart?api-version=2024-07-01
HTTP/1.1
Content-Type: application/json
api-key: {{apiKey}}
```

## 다음 단계

이제 REST 클라이언트에 익숙해지고 Azure AI 검색에 대한 REST 호출을 수행했으므로 벡터 지원을 보여 주는 또 다른 빠른 시작을 시도해 보세요.

[빠른 시작: REST를 사용한 벡터 검색](#)

## 피드백

이 페이지가 도움이 되었나요?

 Yes

 No

[제품 사용자 의견 제공](#) | [Microsoft Q&A에서 도움말 보기](#)

# 빠른 시작: REST API를 사용하여 PowerShell에서 검색 인덱스 만들기

아티클 • 2024. 10. 31.

이 Azure AI 검색 빠른 시작에서는 PowerShell 및 [Azure AI 검색 REST API](#)를 사용하여 검색 인덱스를 만들고, 로드하고, 쿼리하는 방법을 알아봅니다. 이 문서에서는 PowerShell 명령을 대화형으로 실행하는 방법에 대해 설명합니다. 또는 동일한 작업을 수행하는 [PowerShell 스크립트를 다운로드하여 실행](#)할 수 있습니다.

Azure 구독이 아직 없는 경우 시작하기 전에 [체험 계정](#)을 만듭니다.

## 필수 조건

이 빠른 시작에 필요한 서비스와 도구는 다음과 같습니다.

- PowerShell 7.3 이상 ([순차적 및 대화형 단계에 Invoke-RestMethod 사용](#))
- [Azure AI 검색 서비스를 만들거나](#) 현재 구독에서 [기존 서비스를 찾습니다](#). 이 빠른 시작에서는 체험 서비스를 사용할 수 있습니다.

## 검색 서비스 키 및 URL 복사

이 빠른 시작에서 REST 호출에는 모든 요청에 대한 액세스 키와 서비스 URL이 포함됩니다. 검색 서비스는 둘 모두를 사용하여 작성되므로 Azure AI 검색을 구독에 추가한 경우 다음 단계에 따라 필요한 정보를 확보합니다.

- [Azure Portal](#)에 로그인합니다. 검색 서비스의 [개요](#) 페이지에서 다음 URL을 가져옵니다. 엔드포인트의 예는 다음과 같습니다. <https://mydemo.search.windows.net>
- [설정>키](#)를 선택한 다음, 서비스에 대한 모든 권한의 관리자 키를 가져옵니다. 교체 가능한 두 개의 관리자 키가 제공되며, 하나를 롤오버해야 하는 경우 비즈니스 연속성을 위해 다른 하나가 제공됩니다. 개체 추가, 수정 및 삭제 요청 시 기본 또는 보조 키를 사용할 수 있습니다.

모든 요청에서 서비스에 보내는 각 요청마다 API 키가 필요합니다. 유효한 키가 있다면 요청을 기반으로 요청을 보내는 애플리케이션과 이를 처리하는 서비스 사이에 신뢰가 쌓입니다.

## Azure AI 검색에 연결

- PowerShell에서 콘텐츠 형식 및 API 키를 저장하는 `$headers` 개체를 만듭니다. 관리 API 키(`<YOUR-ADMIN-API-KEY>`)를 검색 서비스에 유효한 키로 바꿉니다. 이 헤더는 세션 기간 동안 한 번만 설정하면 되지만 모든 요청에 추가됩니다.

PowerShell

```
$headers = @{
    'api-key' = '<YOUR-ADMIN-API-KEY>'
    'Content-Type' = 'application/json'
    'Accept' = 'application/json' }
```

- 서비스의 indexes 컬렉션을 지정하는 `$url` 개체를 만듭니다. 서비스 이름(`<YOUR-SEARCH-SERVICE-NAME>`)을 유효한 검색 서비스로 바꿉니다.

PowerShell

```
$url = "https://<YOUR-SEARCH-SERVICE-NAME>.search.windows.net/indexes?
api-version=2024-07-01&$select=name"
```

- `Invoke-RestMethod`를 실행하여 GET 요청을 서비스에 보내고 연결을 확인합니다. 서비스에서 다시 보낸 응답을 볼 수 있도록 `ConvertTo-Json`을 추가합니다.

## PowerShell

```
Invoke-RestMethod -Uri $url -Headers $headers | ConvertTo-Json
```

서비스가 비어 있고 인덱스가 없는 경우 결과는 다음 예제와 비슷합니다. 그렇지 않으면 인덱스 정의의 JSON 표현이 표시됩니다.

```
{  
    "@odata.context":  
    "https://mydemo.search.windows.net/$metadata#indexes",  
    "value": [  
        ]  
}
```

## 인덱스 만들기

포털을 사용하지 않는 경우 데이터를 로드하려면 먼저 서비스에 인덱스가 있어야 합니다. 이 단계에서는 인덱스를 정의하고 서비스로 푸시합니다. 이 단계에는 [인덱스 만들기](#) REST API가 사용됩니다.

인덱스의 필수 요소에는 name 및 fields 컬렉션이 포함됩니다. fields 컬렉션은 문서의 구조를 정의합니다. 각 필드에는 사용되는 방법(예: 검색 결과에서 전체 텍스트 검색 가능, 필터링 가능 또는 조회 가능 여부)을 결정하는 이름, 형식 및 속성이 있습니다. 인덱스 내에서 `Edm.String` 형식의 필드 중 하나는 문서 ID에 대한 key로 지정해야 합니다.

이 인덱스는 이름이 `hotels-quickstart`이며 다음 코드에 표시된 필드 정의가 있습니다. 다른 연습 문서에서 사용되는 더 큰 [Hotels 인덱스](#)의 하위 집합입니다. 이 빠른 시작에서는 필드 정의를 잘라 가결하게 만들었습니다.

1. 다음 예제를 PowerShell에 붙여넣어 인덱스 스키마가 포함된 `$body` 개체를 만듭니다

PowerShell

```
$body = @"
{
    "name": "hotels-quickstart",
    "fields": [
        {"name": "HotelId", "type": "Edm.String", "key": true,
 "filterable": true},
        {"name": "HotelName", "type": "Edm.String", "searchable": true,
 "filterable": false, "sortable": true, "facetable": false}.
```

```

        {"name": "Description", "type": "Edm.String", "searchable": true, "filterable": false, "sortable": false, "facetable": false, "analyzer": "en.lucene"},  

            {"name": "Category", "type": "Edm.String", "searchable": true, "filterable": true, "sortable": true, "facetable": true},  

            {"name": "Tags", "type": "Collection(Edm.String)", "searchable": true, "filterable": true, "sortable": false, "facetable": true},  

            {"name": "ParkingIncluded", "type": "Edm.Boolean", "filterable": true, "sortable": true, "facetable": true},  

            {"name": "LastRenovationDate", "type": "Edm.DateTimeOffset", "filterable": true, "sortable": true, "facetable": true},  

            {"name": "Rating", "type": "Edm.Double", "filterable": true, "sortable": true, "facetable": true},  

            {"name": "Address", "type": "Edm.ComplexType", "fields": [  

                {"name": "StreetAddress", "type": "Edm.String", "filterable": false, "sortable": false, "facetable": false, "searchable": true},  

                {"name": "City", "type": "Edm.String", "searchable": true, "filterable": true, "sortable": true, "facetable": true},  

                {"name": "StateProvince", "type": "Edm.String", "searchable": true, "filterable": true, "sortable": true, "facetable": true},  

                {"name": "PostalCode", "type": "Edm.String", "searchable": true, "filterable": true, "sortable": true, "facetable": true},  

                {"name": "Country", "type": "Edm.String", "searchable": true, "filterable": true, "sortable": true, "facetable": true}  

            ]}  

        }  

    ]  

}
"@
```

2. URI를 서비스의 indexes 컬렉션 및 `hotels-quickstart` 인덱스로 설정합니다.

PowerShell

```
$url = "https://<YOUR-SEARCH-SERVICE>.search.windows.net/indexes/hotels-quickstart?api-version=2024-07-01"
```

3. `$url`, `$headers` 및 `$body` 와 함께 명령을 실행하여 서비스에 인덱스를 만듭니다.

PowerShell

```
Invoke-RestMethod -Uri $url -Headers $headers -Method Put -Body $body | ConvertTo-Json
```

결과는 다음 예제와 비슷하며, 여기서는 간결하게 하기 위해 처음 두 필드만 표시되었습니다.

```
{  
    "@odata.context":  
        "https://mydemo.search.windows.net/$metadata#indexes/$entity",  
    "@odata.etag":  "\"0x8D6EDE28CFEABDA\"",  
    "name":  "hotels-quickstart",  
    "defaultScoringProfile":  null,  
    "fields":  [  
        {  
            "name":  "HotelId",  
            "type":  "Edm.String",  
            "searchable":  true,  
            "filterable":  true,  
            "retrievable":  true,  
            "sortable":  true,  
            "facetable":  true,  
            "key":  true,  
            "indexAnalyzer":  null,  
            "searchAnalyzer":  null,  
            "analyzer":  null,  
            "synonymMaps":  ""  
        },  
        {  
            "name":  "HotelName",  
            "type":  "Edm.String",  
            "searchable":  true,  
            "filterable":  false,  
            "retrievable":  true,  
            "sortable":  true,  
            "facetable":  false,  
            "key":  false,  
            "indexAnalyzer":  null,  
            "searchAnalyzer":  null,  
            "analyzer":  null,  
            "synonymMaps":  ""  
        },  
        . . .  
    ]  
}
```

### 💡 팁

확인을 위해 포털에서 **인덱스** 목록을 확인할 수도 있습니다.

## 문서 로드

문서를 푸시하려면 인덱스의 URL 앤드포인트에 대한 HTTP POST 요청을 사용합니다. 이 작업을 위한 REST API는 [문서 인덱싱](#)입니다.

1. 다음 예제를 PowerShell에 붙여넣어 업로드하려는 문서가 포함된 `$body` 개체를 만듭니다.

이 요청에는 두 개의 전체 레코드와 하나의 부분 레코드가 포함됩니다. 부분 레코드는 불완전한 문서를 업로드할 수 있음을 보여 줍니다. `@search.action` 매개 변수는 인덱싱을 수행하는 방법을 지정합니다. 유효한 값은 `upload`, `merge`, `mergeOrUpload` 및 `delete`을 포함합니다. `mergeOrUpload` 동작은 `hotelId = 3`에 대한 새 문서를 만들거나, 이미 있는 경우 내용을 업데이트합니다.

PowerShell

```
$body = @"  
{  
    "value": [  
        {  
            "@search.action": "upload",  
            "HotelId": "1",  
            "HotelName": "Stay-Kay City Hotel",  
            "Description": "The hotel is ideally located on the main commercial artery of the city in the heart of New York. A few minutes away is Time's Square and the historic centre of the city, as well as other places of interest that make New York one of America's most attractive and cosmopolitan cities.",  
            "Category": "Boutique",  
            "Tags": [ "pool", "air conditioning", "concierge" ],  
            "ParkingIncluded": false,  
            "LastRenovationDate": "1970-01-18T00:00:00Z",  
            "Rating": 3.60,  
            "Address":  
                {  
                    "StreetAddress": "677 5th Ave",  
                    "City": "New York",  
                    "StateProvince": "NY",  
                    "PostalCode": "10022",  
                    "Country": "USA"  
                }  
        },  
        {  
            "@search.action": "upload",  
            "HotelId": "2",  
            "HotelName": "Old Century Hotel",  
            "Description": "The hotel is situated in a nineteenth century plaza, which has been expanded and renovated to the highest architectural standards to create a modern, functional and first-class hotel in which art and unique historical elements coexist with the most modern comforts.",  
            "Category": "Boutique",  
            "Tags": [ "pool", "free wifi", "concierge" ],  
            "ParkingIncluded": false,  
            "LastRenovationDate": "1979-02-18T00:00:00Z",  
            "Rating": 3.60,  
            "Address":  
                {  
                    "StreetAddress": "123 Main Street",  
                    "City": "New York",  
                    "StateProvince": "NY",  
                    "PostalCode": "10010",  
                    "Country": "USA"  
                }  
        }  
    ]  
}
```

```
        {
            "StreetAddress": "140 University Town Center Dr",
            "City": "Sarasota",
            "StateProvince": "FL",
            "PostalCode": "34243",
            "Country": "USA"
        }
    },
{
    "@search.action": "upload",
    "HotelId": "3",
    "HotelName": "Gastronomic Landscape Hotel",
    "Description": "The Hotel stands out for its gastronomic excellence under the management of William Dough, who advises on and oversees all of the Hotel's restaurant services.",
    "Category": "Resort and Spa",
    "Tags": [ "air conditioning", "bar", "continental breakfast" ],
    "ParkingIncluded": true,
    "LastRenovationDate": "2015-09-20T00:00:00Z",
    "Rating": 4.80,
    "Address":
    {
        "StreetAddress": "3393 Peachtree Rd",
        "City": "Atlanta",
        "StateProvince": "GA",
        "PostalCode": "30326",
        "Country": "USA"
    }
},
{
    "@search.action": "upload",
    "HotelId": "4",
    "HotelName": "Sublime Palace Hotel",
    "Description": "Sublime Palace Hotel is located in the heart of the historic center of Sublime in an extremely vibrant and lively area within short walking distance to the sites and landmarks of the city and is surrounded by the extraordinary beauty of churches, buildings, shops and monuments. Sublime Palace is part of a lovingly restored 1800 palace.",
    "Category": "Boutique",
    "Tags": [ "concierge", "view", "24-hour front desk service" ],
    "ParkingIncluded": true,
    "LastRenovationDate": "1960-02-06T00:00:00Z",
    "Rating": 4.60,
    "Address":
    {
        "StreetAddress": "7400 San Pedro Ave",
        "City": "San Antonio",
        "StateProvince": "TX",
        "PostalCode": "78216",
        "Country": "USA"
    }
}
]
```

```
}
```

```
"@
```

2. 엔드포인트를 `hotels-quickstart docs` 컬렉션으로 설정하고 인덱스 작업 (`indexes/hotels-quickstart/docs/index`)을 포함합니다.

PowerShell

```
$url = "https://<YOUR-SEARCH-SERVICE>.search.windows.net/indexes/hotels-quickstart/docs/index?api-version=2024-07-01"
```

3. `$url`, `$headers` 및 `$body` 와 함께 명령을 실행하여 문서를 `hotels-quickstart` 인덱스로 로드합니다.

PowerShell

```
Invoke-RestMethod -Uri $url -Headers $headers -Method Post -Body $body  
| ConvertTo-Json
```

결과는 다음 예제와 비슷합니다. [201 상태 코드](#)가 표시됩니다.

```
{  
    "@odata.context":  
    "https://mydemo.search.windows.net/indexes(\u0027hotels-  
    quickstart\u0027)/$metadata#Collection(Microsoft.Azure.Search.V2019_05_  
    06.IndexResult)",  
    "value": [  
        {  
            "key": "1",  
            "status": true,  
            "errorMessage": null,  
            "statusCode": 201  
        },  
        {  
            "key": "2",  
            "status": true,  
            "errorMessage": null,  
            "statusCode": 201  
        },  
        {  
            "key": "3",  
            "status": true,  
            "errorMessage": null,  
            "statusCode": 201  
        },  
        {  
            "key": "4",  
            "status": true,  
            "errorMessage": null,  
            "statusCode": 201  
        }  
    ]  
}
```

```
        "key": "4",
        "status": true,
        "errorMessage": null,
        "statusCode": 201
    }
]
```

## 인덱스 검색

이 단계에서는 [문서 검색 API](#)를 사용하여 인덱스를 쿼리하는 방법을 보여 줍니다.

검색 `$urls`에서는 작은따옴표를 사용해야 합니다. 쿼리 문자열에는 `$` 문자가 포함되며, 전체 문자열을 작은따옴표로 묶으면 이스케이프 처리를 생략할 수 있습니다.

1. 엔드포인트를 `hotels-quickstart docs` 컬렉션으로 설정하고, 쿼리 문자열을 전달하는 `search` 매개 변수를 추가합니다.

이 문자열은 빈 검색(`search=*`)을 실행하여 순위가 없는 임의 문서 목록(검색 점수 = 1.0)을 반환합니다. 기본적으로 Azure AI 검색은 한 번에 50개의 일치 항목을 반환합니다. 구조적으로 이 쿼리는 전체 문서 구조와 값을 반환합니다. `$count=true`를 추가하여 결과에 있는 모든 문서의 수를 가져옵니다.

PowerShell

```
$url = 'https://<YOUR-SEARCH-
SERVICE>.search.windows.net/indexes/hotels-quickstart/docs?api-
version=2024-07-01&search=*&$count=true'
```

2. 명령을 실행하여 서비스에 `$url`을 보냅니다.

PowerShell

```
Invoke-RestMethod -Uri $url -Headers $headers | ConvertTo-Json
```

결과는 다음 출력과 비슷합니다.

```
{
"@odata.context":
"https://mydemo.search.windows.net/indexes(\u0027hotels-
quickstart\u0027)/$metadata#docs(*)",
"@odata.count": 4,
"value": [
{
```

```

        "@search.score": 0.1547872,
        "HotelId": "2",
        "HotelName": "Old Century Hotel",
        "Description": "The hotel is situated in a
nineteenth century plaza, which has been expanded and renovated to the
highest architectural standards to create a modern, functional and
first-class hotel in which art and unique historical elements coexist
with the most modern comforts.",
        "Category": "Boutique",
        "Tags": "pool free wifi concierge",
        "ParkingIncluded": false,
        "LastRenovationDate": "1979-02-18T00:00:00Z",
        "Rating": 3.6,
        "Address": "@{StreetAddress=140 University Town
Center Dr; City=Sarasota; StateProvince=FL; PostalCode=34243;
Country=USA}"
    },
{
    "@search.score": 0.009068266,
    "HotelId": "3",
    "HotelName": "Gastronomic Landscape Hotel",
    "Description": "The Hotel stands out for its
gastronomic excellence under the management of William Dough, who
advises on and oversees all of the Hotel\u0027s restaurant services.",
    "Category": "Resort and Spa",
    "Tags": "air conditioning bar continental
breakfast",
    "ParkingIncluded": true,
    "LastRenovationDate": "2015-09-20T00:00:00Z",
    "Rating": 4.8,
    "Address": "@{StreetAddress=3393 Peachtree Rd;
City=Atlanta; StateProvince=GA; PostalCode=30326; Country=USA}"
},
.
.
.
]
}

```

구문을 이해하기 위해 몇 가지 다른 쿼리 예제를 시도해봅니다. 문자열 검색, 축자 `$filter` 쿼리를 수행하고, 결과 세트를 제한하고, 검색 범위를 특정 필드로 지정하는 등 의 작업을 수행할 수 있습니다.

#### PowerShell

```

# Query example 1
# Search the entire index for the terms 'restaurant' and 'wifi'
# Return only the HotelName, Description, and Tags fields
$url = 'https://<YOUR-SEARCH-SERVICE>.search.windows.net/indexes/hotels-
quickstart/docs?api-version=2024-07-01&search=restaurant
wifi&$count=true&$select=HotelName,Description,Tags'

# Query example 2
# Apply a filter to the index to find hotels rated 4 or higher

```

```
# Returns the HotelName and Rating. Two documents match.  
$url = 'https://<YOUR-SEARCH-SERVICE>.search.windows.net/indexes/hotels-  
quickstart/docs?api-version=2024-07-01&search=*&$filter=Rating gt  
4&$select=HotelName,Rating'  
  
# Query example 3  
# Take the top two results, and show only HotelName and Category in the  
results  
$url = 'https://<YOUR-SEARCH-SERVICE>.search.windows.net/indexes/hotels-  
quickstart/docs?api-version=2024-07-  
01&search=boutique&$top=2&$select=HotelName,Category'  
  
# Query example 4  
# Sort by a specific field (Address/City) in ascending order  
  
$url = 'https://<YOUR-SEARCH-SERVICE>.search.windows.net/indexes/hotels-  
quickstart/docs?api-version=2024-07-01&search=pool&$orderby=Address/City  
asc&$select=HotelName, Address/City, Tags, Rating'
```

## 리소스 정리

본인 소유의 구독으로 이 모듈을 진행하고 있는 경우에는 프로젝트가 끝날 때 여기에서 만든 리소스가 계속 필요한지 확인하는 것이 좋습니다. 계속 실행되는 리소스에는 요금이 부과될 수 있습니다. 리소스를 개별적으로 삭제하거나 리소스 그룹을 삭제하여 전체 리소스 세트를 삭제할 수 있습니다.

가장 왼쪽 창의 **모든 리소스** 또는 **리소스 그룹** 링크를 사용하여 포털에서 리소스를 찾고 관리할 수 있습니다.

무료 서비스를 사용하는 경우 인덱스, 인덱서, 데이터 원본 3개로 제한됩니다. 포털에서 개별 항목을 삭제하여 제한 이하로 유지할 수 있습니다.

## 다음 단계

이 빠른 시작에서는 PowerShell을 사용하여 Azure AI 검색에서 콘텐츠를 만들고 액세스하기 위한 기본 워크플로를 단계별로 실행했습니다. 이러한 개념을 염두에 두고 Azure 데이터 원본의 인덱싱과 같은 고급 시나리오로 이동하는 것이 좋습니다.

[REST 자습서: Azure AI 검색에서 반구조적 데이터\(JSON Blob\) 인덱싱 및 검색](#)

## 피드백

이 페이지가 도움이 되었나요?

 Yes

 No

[제품 사용자 의견 제공](#) | [Microsoft Q&A에서 도움말 보기](#)

# 빠른 시작: Azure Resource Manager 템플릿을 사용하여 Azure AI 검색 배포

아티클 • 2024. 10. 16.

이 문서에서는 Azure Portal에서 ARM(Azure Resource Manager) 템플릿을 사용하여 Azure AI 검색 리소스를 배포하는 과정을 안내합니다.

Azure Resource Manager 템플릿은 프로젝트에 대한 인프라 및 구성을 정의하는 JSON(JavaScript Object Notation) 파일입니다. 이 템플릿은 선언적 구문을 사용합니다. 배포를 만들기 위한 프로그래밍 명령의 시퀀스를 작성하지 않고 의도하는 배포를 설명합니다.

템플릿에 포함된 속성만 배포에 사용됩니다. 네트워크 보안 설정과 같은 추가 사용자 지정이 필요한 경우 서비스를 배포 후 작업으로 업데이트할 수 있습니다. 최소한의 단계로 기존 서비스를 사용자 지정하려면 Azure CLI 또는 Azure PowerShell을 사용합니다. 미리 보기 기능을 평가하는 경우 관리 REST API를 사용합니다.

해당 환경이 필수 조건을 충족하고 ARM 템플릿 사용에 익숙한 경우 Azure에 배포 단추를 선택합니다. 그러면 Azure Portal에서 템플릿이 열립니다.



Deploy to Azure



## 필수 조건

Azure 구독이 없는 경우, 시작하기 전에 [무료 계정](#)을 만드십시오.

## 템플릿 검토

이 빠른 시작에서 사용되는 템플릿은 [Azure 빠른 시작 템플릿](#)에서 나온 것입니다.

```
JSON

{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "metadata": {
    "_generator": {
      "name": "bicep",
      "version": "0.5.6.12127",
      "templateHash": "11257266040777038564"
    }
}
```

```
},
"parameters": {
  "name": {
    "type": "string",
    "maxLength": 60,
    "minLength": 2,
    "metadata": {
      "description": "Service name must only contain lowercase letters, digits or dashes, cannot use dash as the first two or last one characters, cannot contain consecutive dashes, and is limited between 2 and 60 characters in length."
    }
  },
  "sku": {
    "type": "string",
    "defaultValue": "standard",
    "metadata": {
      "description": "The pricing tier of the search service you want to create (for example, basic or standard)."
    },
    "allowedValues": [
      "free",
      "basic",
      "standard",
      "standard2",
      "standard3",
      "storage_optimized_l1",
      "storage_optimized_l2"
    ]
  },
  "replicaCount": {
    "type": "int",
    "defaultValue": 1,
    "maxValue": 12,
    "minValue": 1,
    "metadata": {
      "description": "Replicas distribute search workloads across the service. You need at least two replicas to support high availability of query workloads (not applicable to the free tier)."
    }
  },
  "partitionCount": {
    "type": "int",
    "defaultValue": 1,
    "allowedValues": [
      1,
      2,
      3,
      4,
      6,
      12
    ],
    "metadata": {
      "description": "Partitions allow for scaling of document count as well as faster indexing by sharding your index over multiple search units."
    }
  }
}
```

```

        }
    },
    "hostingMode": {
        "type": "string",
        "defaultValue": "default",
        "allowedValues": [
            "default",
            "highDensity"
        ],
        "metadata": {
            "description": "Applicable only for SKUs set to standard3. You can set this property to enable a single, high density partition that allows up to 1000 indexes, which is much higher than the maximum indexes allowed for any other SKU."
        }
    },
    "location": {
        "type": "string",
        "defaultValue": "[resourceGroup().location]",
        "metadata": {
            "description": "Location for all resources."
        }
    }
},
"resources": [
{
    "type": "Microsoft.Search/searchServices",
    "apiVersion": "2020-08-01",
    "name": "[parameters('name')]",
    "location": "[parameters('location')]",
    "sku": {
        "name": "[parameters('sku')]"
    },
    "properties": {
        "replicaCount": "[parameters('replicaCount')]",
        "partitionCount": "[parameters('partitionCount')]",
        "hostingMode": "[parameters('hostingMode')]"
    }
}
]
}

```

이 템플릿에 정의된 Azure 리소스는 다음과 같습니다.

- [Microsoft.Search/searchServices](#): Azure AI 검색 서비스 만들기

## 템플릿 배포

다음 이미지를 선택하고 Azure에 로그인하여 템플릿을 업니다. 이 템플릿은 Azure AI 검색 리소스를 만듭니다.



Deploy to Azure



포털에는 매개 변수 값을 쉽게 입력할 수 있는 양식이 표시됩니다. 일부 매개 변수는 템플릿의 기본값으로 미리 채워져 있습니다. 구독, 리소스 그룹, 위치 및 서비스 이름을 입력해야 합니다. [AI 보강](#) 파이프라인에서 Azure AI 서비스를 사용하려면(예: 텍스트의 이진 이미지 파일 분석) Azure AI 검색 및 Azure AI 서비스를 모두 제공하는 위치를 선택합니다. 두 서비스가 AI 보강 워크로드에 대한 동일한 지역에 있어야 합니다. 양식을 모두 작성한 후에는 사용 약관에 동의하고 구매 단추를 선택하여 배포를 완료해야 합니다.

## Azure Search service

Azure quickstart template

### TEMPLATE



101-azure-search-create

1 resource



Edit template



Edit paramet...



Learn more

### BASICS

Subscription \*

▼

Resource group \*

▼

Create new

Location \*

▼

### SETTINGS

Name \* ⓘ

Sku ⓘ

▼

Replica Count ⓘ

Partition Count ⓘ

▼

Hosting Mode ⓘ

▼

Location ⓘ

### TERMS AND CONDITIONS

[Template information](#) | [Azure Marketplace Terms](#) | [Azure Marketplace](#)

By clicking "Purchase," I (a) agree to the applicable legal terms associated with the offering; (b) authorize Microsoft to charge or bill my current payment method for the fees associated with the offering(s), including applicable taxes, with the same billing frequency as my Azure subscription, until I discontinue use of the offering(s); and (c) agree that, if the deployment involves 3rd party offerings, Microsoft may share my contact information and other details of such deployment with the publisher of that offering.

 I agree to the terms and conditions stated above

## 배포된 리소스 검토

배포가 완료되면 포털에서 새 리소스 그룹 및 새 검색 서비스에 액세스할 수 있습니다.

# 리소스 정리

다른 Azure AI 검색 빠른 시작과 자습서는 이 빠른 시작을 기반으로 작성됩니다. 후속 빠른 시작과 자습서를 계속 진행할 계획이라면 이 리소스를 그대로 두는 것이 좋습니다. 이 리소스가 더 이상 필요 없으면 리소스 그룹을 삭제해도 됩니다. 그러면 Azure AI 검색 서비스 및 관련 리소스가 삭제됩니다.

## 다음 단계

이 빠른 시작에서는 ARM 템플릿을 사용하여 Azure AI 검색 서비스를 만들고 배포의 유효성을 검사했습니다. Azure AI 검색 및 Azure Resource Manager에 대해 자세히 알아보려면 아래 문서를 계속 진행하세요.

- [Azure AI 검색 개요](#)를 확인합니다.
- 검색 서비스에 대한 [인덱스를 만듭니다.](#)
- 포털 마법사를 사용하여 [데모 앱을 만듭니다.](#)
- 데이터에서 정보를 추출하는 [기술 세트를 만듭니다.](#)

---

## 피드백

이 페이지가 도움이 되었나요?

 Yes

 No

[제품 사용자 의견 제공](#) | [Microsoft Q&A에서 도움말 보기](#)