American Impact Review

Theoretical Article

# Search Planning Based on the Type of Storage: Searching for Composite Keys in LLM-Based Corporate Knowledge Systems

Mykola Khoroshevskyi*

*Principal Software Engineer, Sigma Software, Altea, Spain*

* Corresponding author: mykola.khoroshevskyi@icloud.com

OPEN ACCESS

**Citation:**
Mykola Khoroshevskyi (2026). Search Planning Based on the Type of Storage: Searching for Composite Keys in LLM-Based Corporate Knowledge Systems. *Am. Impact Rev.*
10.66308/air.e2026072

**Received:** February 19, 2026
**Accepted:** March 4, 2026
**Published:** March 6, 2026

**DOI:**
10.66308/air.e2026072

**ISSN:** 3071-124X



## Abstract

The article describes the planning of search in corporate knowledge systems using large language models, taking into account the structure of the data warehouse. Special attention is paid to the search for objects that are identified by composite keys. To do this, you need to take into account the data schema, the relationships between objects, the available indexes, and the order in which sources are accessed. The article discusses various approaches to search: lexical, vector, hybrid, federated, and multi-stage. The architecture of LLM-guided search and the algorithm for processing complex queries are also presented. It is shown that large language models can be used to interpret queries, select search actions, and refine subsequent operations. However, information about the data structure must come from the metadata of the repository. To evaluate the proposed solutions, BEAVER and HERB enterprise suites are being considered, which include verification of work with structured and heterogeneous data sources.

**Keywords:** corporate knowledge systems, large language models, LLM, composite key, information search, search planning, storage structure, hybrid search, multi-stage search, Retrieval-Augmented Generation, RAG, corporate data

## Relevance of the study

## Relevance of the study

With the development of large language models, the information retrieval and processing capabilities in corporate knowledge systems are improving. However, since corporate data is often stored in different locations and has different structures, only semantic search or the standard Retrieval-Augmented Generation (RAG) approach is not enough to get accurate results.

Finding objects identified by composite keys, i.e., multiple related values can be particularly difficult. In such cases, it is important to consider the structure of the storage, the composition of the identifiers, and the order in which the data is accessed. Large language models can be used not only to generate responses to queries but also to plan the search. They help you select the source, determine the necessary parameters, and consistently refine the request, which greatly speeds up the process.

In this regard, it becomes very relevant to study search-planning methods that take into account the structure of the repository and use large language models to search for composite keys. This approach is aimed at improving the accuracy and efficiency of access to diverse data in corporate knowledge systems.

## The purpose of the study

## The purpose of the study

The purpose of this study is to test the approach to designing the search for composite keys using a large language model, taking into account the structure of the corporate data warehouse. It also defines the main stages, architectural components, and evaluation criteria for such a search.

## Materials and research methods

## Materials and research methods

The research materials covered information about modern corporate search methods, the Azure AI Search agent's search architecture, multi-stage query processing mechanisms, and the characteristics of the BEAVER and HERB corporate datasets.

The paper used methods of analysis and systematization of scientific and technical materials, a comparative analysis of approaches to information search, a structural and functional analysis of the search engine's architecture, and an analysis of the sequence of search operations. In order to determine possible criteria for experimental verification, accuracy, completeness, and F1 indicators were considered in evaluating the search for database elements.

## The results of the study

## The results of the study

Modern corporate knowledge systems use various approaches to information retrieval, each of which is suitable for working with a specific type of data and the nature of user queries. Lexical search is focused on exact matches of terms, vector search is focused on semantic proximity, and hybrid and multi-stage methods combine several data extraction and processing mechanisms. The main characteristics of these approaches are presented in Table 1 for clarity.

**Table 1.** Basic approaches to information retrieval in corporate knowledge systems

| Approach | The basic principle | Scope of application |
|---|---|---|
| Lexical search | Comparison of query terms with the content of the indexed text | Search for exact terms, names, and identifiers |
| Vector search | Search by similarity of vector representations | Search for semantically similar information |
| Hybrid Search | Combining lexical and vector search | A combination of precise and semantic matching |
| Re-ranking | Re-evaluation of previously found documents | Clarifying the order of results after the initial search |
| Federated search | Using multiple independent sources | Search in distributed corporate knowledge sources |
| Multi-stage search using LLM | Sequential execution of several search steps | Processing complex queries that require multiple operations |

*A source: compiled by the author on the basis of materials [5], [9].*

The presented approaches differ both in the way the query is compared with the data and in the organization of the search process. In corporate systems, where there are several interconnected sources, not only the

quality of a single search engine is of particular importance, but also the ability to determine the sequence of data access. This opens the way to considering search planning that takes into account the structure of the repository.

Storage-oriented search planning requires the use of information about the data schema, keys, relationships between objects, and available indexes. In relational databases, the primary key can consist of several columns, and the effectiveness of searching in a multi-column index directly depends on which fields are involved in the query. Therefore, the storage structure directly affects the choice of data access method.

When processing queries in natural language, the key step is to identify relevant tables, fields, and relationships before starting the main search. A large language model can be used to interpret the query and determine the sequence of actions; however, information about the data structure must come from the actual metadata of the repository.

When searching for a composite key, the system first determines the known values, then, using the associated data finds the missing components and only then performs an accurate search for the object. This approach allows you to take into account the real structure of corporate storage and significantly reduce the number of unnecessary requests.

The architecture of enterprise search driven by large language models (LLM) assumes separation of functions between a large language model, a query planning mechanism, and direct data access tools. In the Azure AI Search architecture, the knowledge base is responsible for planning and executing search queries, and the agent gets access to it through the appropriate search tool. Knowledge sources are linked to search indexes that contain information available for retrieval. The main components of such a search interact in a certain sequence, as shown in Figure 1.

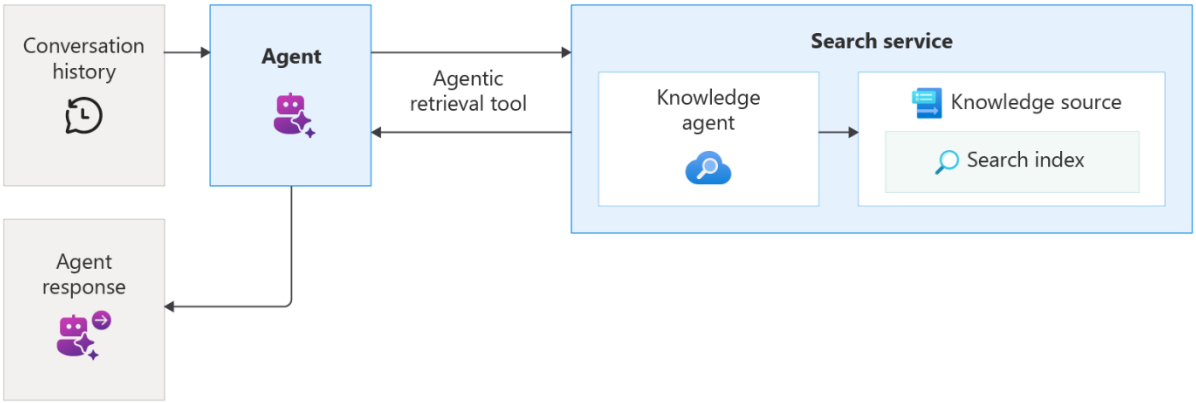


**Figure 1.** Agent search architecture in Azure AI Search [8]

As can be seen from the diagram in Figure 1, there is a search management layer between the user query and the search index. This level is responsible for analyzing the request, determining the necessary actions, and accessing the appropriate data sources. This architecture differs from a simpler scheme in which one query is immediately transmitted to the search engine.

When a complex query is being processed, the language model can split it into several simpler subqueries. In Azure AI Search agent-based search, this mechanism is used as follows: individual subqueries are performed based on available sources, and then the results found are combined and ranked. This allows you to process requests that require information from several parts of the corporate knowledge base to be answered [6].

In corporate systems, information access control is an important component of the architecture. The search engine must not only find relevant data but also take into account user permissions. Azure AI Search

implements access control mechanisms based on Microsoft Entra ID, as well as filtering results during query execution and network restrictions.

The LLM-driven search architecture (Language model) is especially useful for queries that cannot be performed using a single data source. In such a system, the language model is responsible for interpreting the query and determining the sequence of actions, and specialized search tools directly search for information. Separation of functions allows you to take advantage of LLM when working with natural language, while maintaining access to actual data within existing corporate repositories.

The multi-step search algorithm begins with the analysis of the original query. In Azure AI Search agent-based search, the large language model is able to break down a complex query into several simpler subqueries. These subqueries are executed in parallel, after which the found materials are semantically re-ranked and combined into a common result. This approach is designed to address issues that are difficult to cover in a single search query.

Multistep search can be performed iteratively. In accordance with Microsoft's agentic RAG recommendations, the agent analyzes intermediate results and, if there is insufficient information, executes an additional query with specified conditions. Thus, the execution plan does not always remain unchanged after the first access to the source: subsequent actions may depend on the data already received. The general sequence of such processing is shown in Figure 2.

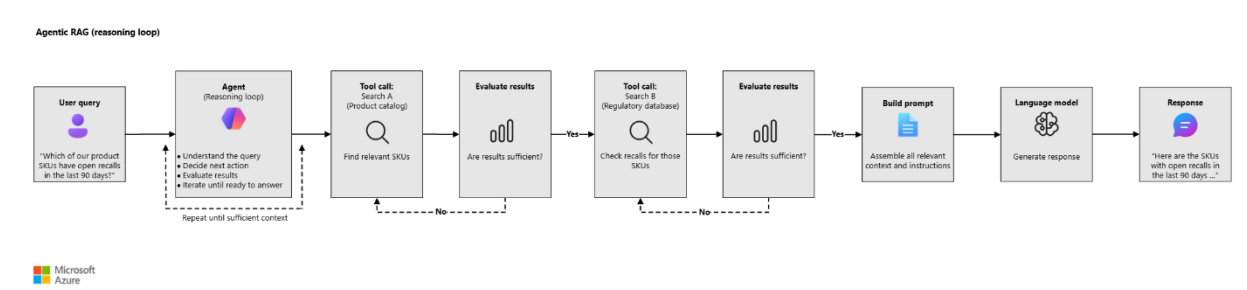


**Figure 2.** The sequence of request processing in the agent-based RAG system [4]

As can be seen in Figure 2, the search process is a separate stage in the agent's work cycle. The agent can choose an appropriate way to receive information, process the received data, and determine whether it needs to access the data again. This principle corresponds to the ReAct approach, in which the language model alternates the stages of reasoning, acting, and obtaining results from the external environment, using new information to make further decisions [10].

After completing the subqueries, you need to combine and organize the results. Azure AI Search uses the Reciprocal Rank Fusion method to combine multiple ranked lists. The semantic mechanism then re-evaluates the most relevant results. The parameters of these operations, which can be found in the documentation, are shown in Table 2.

**Table 2.** Basic parameters for processing search results in Azure AI Search

| Operation | Confirmed characteristic | Assignment during the search process |
|---|---|---|
| Reformulating the request | Up to ten variants of the original query can be created | Expanding the search coverage and taking into account different formulations of the user query |
| Combining the results | The Reciprocal Rank Fusion method is used to combine several ranked lists | Combining the results obtained by different search engines |
| Of the RRF calculation | The score is calculated using the formula 1 / (rank + k), where k is the constant specified in the documentation, equal to 60 | Accounting for the position of a document in several ranked lists |
| Semantic re-ranking | At this stage, the first 50 results of the initial search are used | Increasing the relevance of the final sequence of documents found |
| Generating agent-based search results | The result may include combined content, links to sources, and information about completed actions | Transmitting the final information to the user, indicating the sources used |

*A source: compiled from the data [1], [7].*

Thus, the process, confirmed by open sources, involves receiving and analyzing a query, if necessary decomposing it, performing one or more searches, evaluating the intermediate results, and, if there is a lack of information, conducting additional searches. The found materials are then combined and re-ranked to create a set of data that can be used directly by the system or passed on to a language model for generating a response with links to relevant sources.

It is advisable to conduct an experimental evaluation of a search process driven by a large language model, not only based on the correctness of the final answer, but also on individual stages of working with the structure of the repository. This principle is applied in the BEAVER dataset, which contains 9,128 question-SQL query pairs, 812 tables from three corporate repositories, and 19 subject areas. This dataset separately evaluates the search for necessary tables, definition of connection keys, matching of fields, use of subject knowledge, and decomposition of queries.

Table 3 shows the main characteristics of the storages used to create the BEAVER dataset.

**Table 3.** Characteristics of corporate data warehouses used in BEAVER

| Storage | DBMS | Number of tables | Number of columns | Main content |
|---|---|---|---|---|
| DW | Oracle | 97 | Number of columns | Educational systems and facilities management |
| NW | MySQL | 366 | Number of columns | Computing infrastructure, virtualization, security, and access control |
| SP | MySQL | 349 | Number of columns | Management of residential facilities, check-in, deliveries and events |

*A source: compiled according to the research data [2].*

To evaluate the effectiveness of the proposed approach, the most appropriate indicators are F1 for searching tables, F1 for determining join keys, F1 for matching columns, and the final query accuracy. In the BEAVER

system, the accuracy of query execution is determined by the degree to which the result matches the reference SQL query. F1 for the connection keys is calculated based on a comparison of the found and reference connection conditions of the tables. It should be noted that compound keys and composite primary keys are different concepts. Therefore, BEAVER metrics cannot be directly used to evaluate the search for composite keys. To carry out such an assessment, special marking of the correct composite identifiers is required.

The need to verify a multi-stage search process is particularly evident in corporate settings, where information about a single entity is distributed across several sources. An example of such a task can be seen in the HERB dataset, as shown in Figure 3.

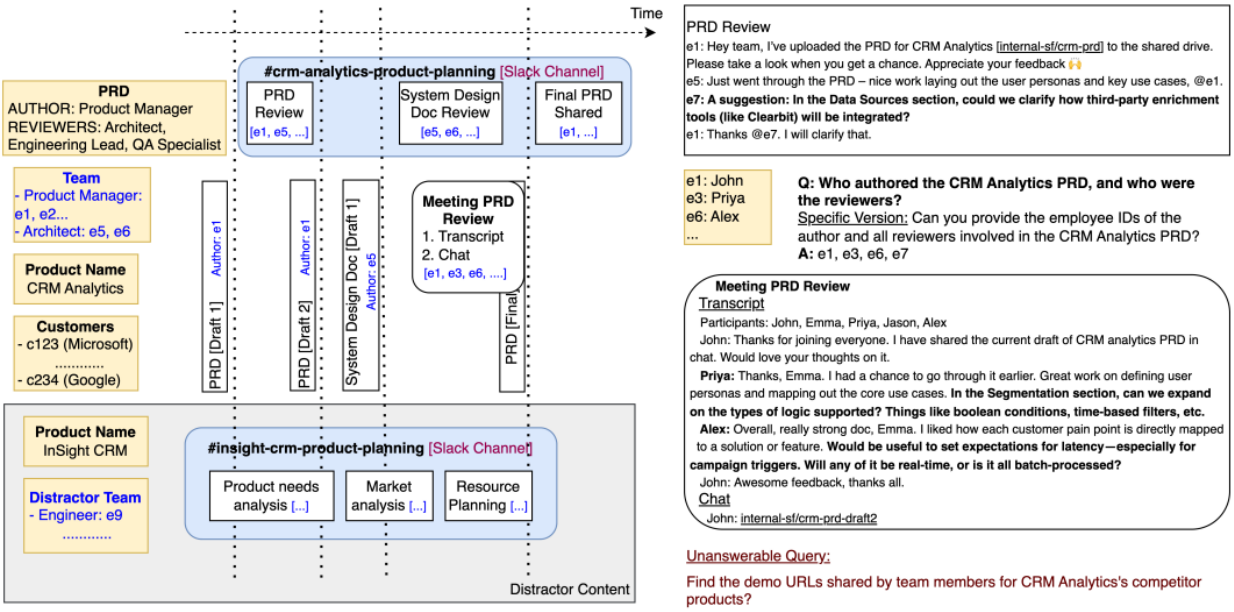


**Figure 3.** An example of a workflow with information distribution between corporate sources in the HERB suite [3]

As can be seen from Figure 3, responding to a request may require sequential access to several sources of information: a document, Slack messages, a transcript of a meeting, and structured data about employee IDs. The HERB kit also contains distracting materials and queries for which the necessary information is missing. The set contains a total of 815 queries with responses and 699 queries without sufficient data. This allows you to test not only the system's ability to find information but also its ability to correctly stop searching if there is no supporting information.

Thus, when testing the proposed method, it is necessary to separate the quality of the query structure formation, the correctness of the selection and combination of data warehouse elements, the completeness of the information found, and the accuracy of the final result. This approach allows you to determine at what stage an error occurs, rather than evaluating the system based solely on the final response. This methodology corresponds to modern standards of detailed assessment of corporate search.


**Mykola Khoroshevskyi**

Principal software engineer, Sigma Software,

Altea, Spain

## Conclusions

The conducted analysis allows us to conclude that in corporate knowledge systems, information search should be carried out taking into account the actual structure of the repository, and not only on the basis of the semantic correspondence of the user query to the found documents. When using composite keys, special attention should be paid to information about fields, relationships, and identifiers, as they determine the order of access to data. In such a system, a large language model can perform the functions of interpreting a query, dividing a task into parts, and selecting subsequent search steps. Direct information acquisition should be carried out by means of corporate storage.

The proposed approach includes sequential query analysis, performing one or more search steps, evaluating intermediate results, further refining the search if necessary, as well as combining and re-ranking the found materials. To conduct an experimental evaluation of this method, attention should be paid to several key aspects: the correctness of the definition of the query structure, the selection and relationship of storage elements, the completeness of the data obtained, and the correctness of the result. This separation helps to determine at what stage an error occurs and comprehensively assess the quality of LLM-managed search in the corporate knowledge system.

## References


1. Agentic retrieval in Azure AI Search // Microsoft Learn: [site]. - URL: https://learn.microsoft.com/en-us/azure/search/agentic-retrieval-overview?tabs=quickstarts.

2. Chen P.B., Yang D., Li W., Wenz F., Zhang Y., Tatbul N., Cafarella M., Demiralp Ç., Stonebraker M. BEAVER: An Enterprise Benchmark for Text-to-SQL // arXiv. - 2024. - arXiv:2409.02038. - DOI: 10.48550/arXiv.2409.02038. - URL: https://arxiv.org/abs/2409.02038.

3. Choubey P. K., Peng X., Bhagavath S., Huang K.-H., Xiong C., Wu C.-S. Benchmarking Deep Search over Heterogeneous Enterprise Data // Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track. - Suzhou, China: Association for Computational Linguistics, 2025. - pp. 501-517. - DOI: 10.18653/v1/2025.emnlp-industry.34. - URL: https://aclanthology.org/2025.emnlp-industry.34/.

4. Develop an agentic RAG solution // Microsoft Learn: [website]. - URL: https://learn.microsoft.com/en-us/azure/architecture/ai-ml/guide/rag/rag-agentic.

5. Hybrid search using vectors and full text in Azure AI Search // Microsoft Learn: [site]. - URL: https://learn.microsoft.com/en-us/azure/search/hybrid-search-overview.

6. Retrieval-augmented generation (RAG) in Azure AI Search // Microsoft Learn: [site]. - URL: https://learn.microsoft.com/en-us/azure/search/retrieval-augmented-generation-overview?tabs=videos.

7. Semantic ranking in Azure AI Search // Microsoft Learn: [website]. - URL: https://learn.microsoft.com/en-us/azure/search/semantic-search-overview.

8. Tutorial: Build an end-to-end agentic retrieval solution using Azure AI Search // Microsoft Learn: [site]. - URL: https://learn.microsoft.com/en-us/azure/search/agentic-retrieval-how-to-create-pipeline.

9. What is hybrid search? How it works and when to use it // elastic: [site]. - URL: https://www.elastic.co/what-is/hybrid-search.

10. Yao S., Zhao J., Yu D., Du N., Shafran I., Narasimhan K., Cao Y. ReAct: Synergizing Reasoning and Acting in Language Models // arXiv. - 2022. - arXiv:2210.03629. - DOI: 10.48550/arXiv.2210.03629. - URL: https://arxiv.org/abs/2210.03629.