# On Data Spaces for Retrieval Augmented Generation

Felix Hermsen [1, 2], Lasse Nitz [1, 2], Mehdi Akbari Gurabi [1, 2], Roman Matzutt [1], and Avikarsha Mandal [1]

**Abstract:** Large Language Models (LLMs) have revolutionized knowledge retrieval from natural language queries. However, LLMs still face challenges regarding the creation of domain-specific and accurate answers. Recently, Retrieval Augmented Generation (RAG) architecture has been proposed as one approach to addressing these challenges. While current research focuses on optimizing document retrieval and augmenting the initial query accordingly, we identify untapped potentials of RAG to retrieve knowledge from heterogeneous data sources via data spaces. In this work, we investigate three conceptual integration scenarios between RAG and data spaces. Our findings indicate that given the data space extended RAG, could provide domain-specific information retrieval with diverse data sources. However, solutions to mitigate unintended information leakage require further consideration.

**Keywords:** Data Spaces, Large Language Models, Retrieval Augmented Generation, Data Sharing

## 1 Introduction

The remarkable performance of *Large Language Models (LLMs)* in a variety of domains has attracted the attention of both academia and industry alike. For example, LLMs revolutionized knowledge retrieval from natural-language queries [BLL23]. However, LLMs continue to face challenges as well. For instance, LLMs remain prone to *hallucination*, i.e., the accidental generation of fabricated or partially erroneous answers that occur as the LLM may miss the correct context and generally relies on statistical models to predict how to continue its answer. This unwanted behavior is particularly pressing in the context of highly domain-specific queries or the utilization of open-source LLMs, such as Llama3 [To23], which often have to rely on fewer training data than their commercial counterparts.

Recently, *Retrieval Augmented Generation (RAG)* [Le20] has been proposed as a means to dynamically add further information and context to a query to mitigate the model's knowledge gap and, thus, hallucination. Current research on RAG focuses on performance improvements by optimizing the data retrieval from the knowledge base and the augmentation step [Ga23]. However, these efforts focus on improving the RAG approach itself and do not question the implicit reliance on *individual, isolated knowledge bases* for data retrieval, which essentially reinforce the notion of *data silos*.

[1] Fraunhofer FIT, DSAI, Schloss Birlinghoven, 53757 Sankt Augustin, Germany,
felix.hermsen@fit.fraunhofer.de, https://orcid.org/0009-0001-0117-3902;
lasse.nitz@fit.fraunhofer.de, https://orcid.org/0000-0002-3131-7444;
mehdi.akbari.gurabi@fit.fraunhofer.de, https://orcid.org/0000-0002-1734-8367;
roman.matzutt@fit.fraunhofer.de, https://orcid.org/0000-0002-4263-5317;
avikarsha.mandal@fit.fraunhofer.de, https://orcid.org/0000-0002-8641-7207
[2] RWTH Aachen University, Information Systems & Databases, Ahornstr. 55, 52074 Aachen, Germany,

doi:10.18420/inf2024_57

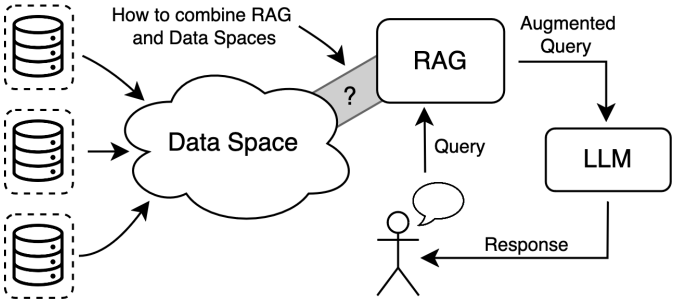


Fig. 1: Concept of data space-extended RAG

Instead, in this work we identify a tremendous untapped potential of RAG to consolidate specialized knowledge from *heterogeneous data sources* maintained by *multiple, independent knowledge owners*. *Data spaces* promise to provide a suitable infrastructure due to their goal of establishing global ecosystems for fine-grained data exchanges [CST22]. In particular, the combination of RAG and data spaces depicted in Fig. 1 could allow users to query over high-quality data maintained by experts of different domains, or combine data sources of competing data owners. However, data spaces focus on enabling data consumers to *find* and *interpret* structured data from remote data sources via standardized interfaces while retaining the data owner's sovereignty over their data.

In this context, we assess the feasibility of bridging these two ecosystems, i.e., strive for a *data space-extended RAG (DS-RAG)* architecture. Namely, we present how RAG systems can benefit from an integration with data spaces and which additional challenges arise in this context. Our discussion reveals that the RAG use case aligns well with the roles defined for data spaces. However, DS-RAG mandates to pay special attention concerning which trust domain to integrate RAG into, as RAG and LLMs may reintroduce challenges regarding unintended information leakage.

**High-Level Data Space Architecture:** On a high level, data spaces are described as a federated, open infrastructure designed for sovereign data sharing, which operates based on common policies, rules, and standards to facilitate cooperation among participants [Ot22]. It allows for interoperability and secure data exchange among different entities. Data sovereignty is a concept that has emerged in this context. It refers to the right of the data owner to maintain control over their data [Ak24]. One approach to achieving data sovereignty is through usage control, which involves introducing and enforcing restrictions on potential data handling. It is a generalized form of traditional access control and focuses on *how*, *where*, or *why* the data is used, in addition to *who* has access. The data provider defines the usage policies, and the usage control mechanism enforces them [JD22]. A data space as defined by Otto *et al.* [Ot22] is a federated data sharing platform that consists of three different roles. The first role is the data provider (DP), while the second role is

the data consumer (DC). Data is transmitted directly from the DP to the DC, bound by a prearranged contract (i.e., policy) that specifies the usage terms of the data, agreed upon beforehand. However, this data exchange is supported by the third role, which is referred to as the federator (F) and provides the necessary trust between parties through additional services. For instance, this role provides services like data brokering, cataloging for data compensation and data sovereignty for privacy and security assurances [Ot22]. Since the data spaces are generally decentralized and federated, some of the federator functionalities can be performed by either the DP or DC. To ensure trust between parties, trusted software must be used. The trusted software that facilitates this process is called a connector. Thus, DC and DP communicate and facilitate trusted data exchange through data space connectors.

**Basic RAG Architecture:** In the basic RAG architecture as detailed by Gao et al. [Ga23] there are two different roles. A user with a natural language prompt and a service provider that possesses a database, an index database and access to a LLM. The index database stores vector encodings of the initial database and is used for efficient search. On receiving a user prompt, the service provider performs a semantic similarity search on the index database to find relevant data for the given prompt. After that, the corresponding real data (context) is merged with the query by generating a new prompt, which optimizes LLM compatibility. This step is commonly referred to as *augmentation process*. Finally, the augmented prompt is sent to the LLM which returns the response to the service provider, who then subsequently sends it to the user.

## 2 Integration Concept

In this section, we present our ideas on how to combine RAG and data spaces. Looking at the high-level view of data spaces and the basic RAG architecture, we identified three distinct integration concepts. They vary in complexity and guarantee the data provider and data customer different levels of control over the exchanged data within the bounds of policy agreements in the data space.

### 2.1 Integration Concept 1 - RAG Facilitated by the Data Consumer

The first and most straightforward idea to use a data space for RAG is to build up a centralized database of relevant data on the DC side and then run the entire RAG architecture on the DC side. In more detail, the DC uses the data brokering functionality of the data space to retrieve a list of potential data providers. After that, the DC contacts each DP and negotiates a policy to retrieve data for the purpose of using it for RAG. The negotiation could be optimized by adding policies specially designed for RAG to the policy store of the data space. This could reduce or even eliminate the actions a human has to take during the negotiation process. After completing all negotiations, the DC retrieves all the data and computes an index

database for efficient search. This process should be independent of a query instance, since retrieving an entire database and building an index for a single query is associated with prohibitive computational effort. The rest of the concept follows the basic RAG model for a single service provider. The advantage of this approach is that no additional functionalities need to be added to the data space. Furthermore, a DP does not have to provide any specific input for RAG. This makes the approach versatile and simple to implement. In addition, since all the functionalities on the DC side are monitored by the DC connector, breaches of the specified policy can be detected. Thus the data provider has full transparency over its data. However, data from multiple DPs has to be merged at the DC and has to remain there for an extended period of time when the DC wants to provide a specialized RAG service to users. As this might contradict the initial data space idea and could discourage DPs from engaging in data exchange, we aim to avoid centralizing any data in the next integration concept.

### 2.2 Integration Concept 2 - Index Database Maintained by the Data Provider

While the last integration concept assumes that the index database is constructed by the DC, this integration concept assumes that data from multiple sources cannot be centralized at the DC's side. Thus, the idea is that each DP maps its own data to an index database and keeps it local. The concept is illustrated by the blue box in Fig 2. In this case, to facilitate the semantic similarity search, the DC must forward embedded user queries to each DP. The embedding of the query could be integrated into the augmentation component to reduce the number of modules. In addition, we assume that all DP use the same embedding model to create their respective local index database. After this, each data provider performs a similarity search over its data against the embedded query and forwards the similarity values to a federated search component that ranks the retrieved results in a privacy-preserving way, retrieves the top $n$ results and forwards them to the DC. For private ranking, a trusted third party could be used or privacy-preserving technologies like functional encryption [BSW11] or differential privacy [Dw06]. Finally, after receiving the relevant context for the query, the DC augments the query and forwards it to its local LLM to retrieve the answer for the user.

As mentioned before, the benefit of this approach in comparison to the first is that the DC data does not have to be centralized anymore. In addition, the retrieved data does not need to be stored anymore and could be deleted after the LLM output is obtained. Additionally, the retrieved context could be buffered for a period of time for frequent queries to reduce the load of the system. The time period of the deletion could be managed inside the policy and monitored by the connector infrastructure. However, the embedded query needs to be sent to each data provider, potentially revealing sensitive information about the user. Furthermore, DP data is still sent to the DC, potentially restricting use cases if the data is too sensitive to share even in an anonymized fashion. For instance, a query about supply chain data should not return confidential supplier information; instead, it should provide only general conclusions about the overall supply chain.

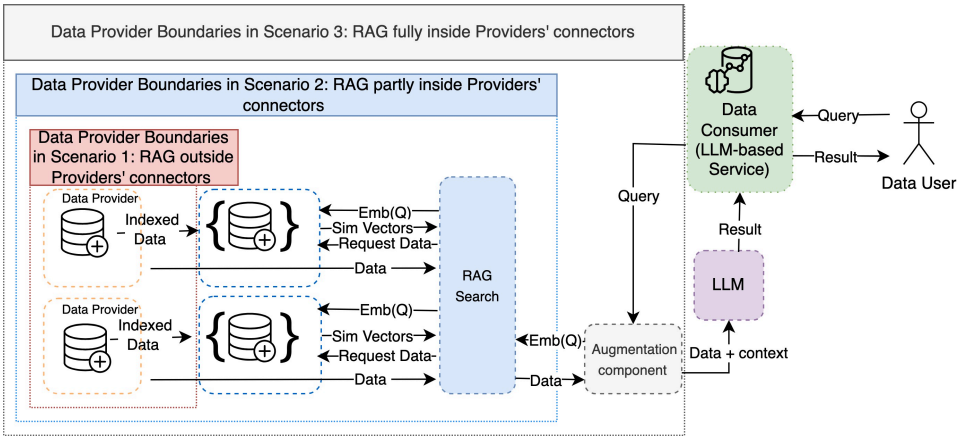


Fig. 2: This conceptual overview presents the three data space-extended RAG integration concepts discussed in this paper. The figure depicts which part of the architecture the data provider controls in which concept. It should be noted that everything outside the respective colored box is assumed to be controlled by the data consumer and inside the box by the data provider. Brackets around a database indicate an index database, while arrows depict a data flows.

### 2.3 Integration Concept 3 - LLM Output Delivered by the Data Provider

Building on top of the second integration concept, the DP might not want to share any information with the DC. The concept is illustrated by the grey box in Fig. 2. Therefore, all the RAG functionality needs to take place within the data space or the DP's connector. In that case, a federator provides the RAG search, augmentation and the LLM. Note that each component can be managed by a different federator. The DC only provides an interface for user queries to a DS-RAG, negotiates access and policies, filters out policy violating queries, and forwards the query data to the correct data space components. As for the second concept, the federated index search is the same. However, after receiving the necessary context a federator also augments the context and sends it to an LLM.

## 3 Discussion and Potential Challenges

We now investigate the three different scenarios of combining RAG architecture with data spaces and look into respective integration challenges. To create DS-RAG, the first approach gave the data consumer the most control. However, as pointed out, some data providers might not want to share data with an entity that centralizes data from different providers. Consequently, the second option, which gives greater control to the data provider, was introduced. The additional features include the sharing of only relevant data and the ability to delete data after the LLM result is obtained through the use of data space functionalities. Nevertheless, the data consumer still obtains data and retains certain control over the data

usage. The downside of that approach for the data consumer is that they need to share their user queries with all data providers that might contain sensitive information like intellectual property. This approach strikes a balance between the data provider and the data consumer in terms of data control. Finally, we introduced the third approach that gives full control to the data providers by not sharing any data except the final LLM result with the data consumer. As such the data consumer just acts as an interface to a data space with a fully integrated RAG component. However, from data user's (the one using LLM-based service) perspective, the trust boundary must be extended from the data consumer to all involved data providers as the RAG search will take place at the data providers' premise.

While these three approaches addressed some data usage concerns, some challenges remain unaddressed. For instance, giving LLM-generated responses with data space knowledge to users outside the data space infrastructure might be concerning. The is because, the generated responses by the LLM might contain retrieved data, meaning the DP and DC lose control over the data once the LLM response has left the ecosystem. Thus, DS-RAG might only be applicable to data space participants. In addition, policy negotiation for DS-RAG could turn out to be complicated in practice. For instance, defining which data is too sensitive for RAG and which is not could be a challenge. Furthermore, defining what to do with sensitive data to actually enable it to be useful for RAG is another challenge.

For future work, one interesting research direction is to develop a specific integration concept for one data space based on integration concepts presented in this work. Possible options are the Energy Data (EDS) [Be22] or Mobility Data Space (MBS) [PDR22], which use the underlying IDS architecture [PSB22]. For instance, one of the early demonstration projects of the EDS concerns the Electricity Grid Stability [Be22]. One could integrate DS-RAG into the EDS to provide a report of the electrical grid stability. In fact, the EDS and MBS list several interesting demonstration projects where DS-RAG could be an asset. Furthermore, design of specific DS-RAG policies and examination of usage control technologies can aid directly in implementing DS-RAG. Furthermore, one could also investigate the resolution time for performing each query imposed by introducing data space-specific components with RAG architecture. This is especially relevant for interactive information retrieval. In addition, considering RAG architectures beyond the basic one, such as modular RAG [Ga23], may allow for utility gains from good context.

## Acknowledgements

This work has been funded by the German Federal Ministry of Education and Research (BMBF) under funding reference number 01IS21100A and by the Union's Horizon Europe Framework Programme project TANGO (Grant Agreement No. 101070052).

## References


[Ak24] Akbari Gurabi, M. et al.: Towards Privacy-Preserving Machine Learning in Sovereign Data Spaces: Opportunities and Challenges. In (Bieker, F. et al., eds.): Privacy and Identity Management. Sharing in a Digital World. Springer Nature, pp. 158–174, 2024.

[Be22] Berkhout, V. et al.: Energy Data Space. In (Otto, B.; ten Hompel, M.; Wrobel, S., eds.): Designing Data Spaces. Springer, pp. 329–341, 2022.

[BLL23] Bommasani, R.; Liang, P.; Lee, T.: Holistic Evaluation of Language Models. Annals of the New York Academy of Sciences 1525 (1), pp. 140–146, 2023.

[BSW11] Boneh, D.; Sahai, A.; Waters, B.: Functional Encryption: Definitions and Challenges. In: Theory of Cryptography. Springer, pp. 253–273, 2011.

[CST22] Curry, E.; Scerri, S.; Tuikka, T.: Data Spaces: Design, Deployment and Future Directions. Springer Nature, 2022.

[Dw06] Dwork, C.: Differential Privacy. In (Bugliesi, M. et al., eds.): Automata, Languages and Programming. Springer, pp. 1–12, 2006.

[Ga23] Gao, Y. et al.: Retrieval-Augmented Generation for Large Language Models: A Survey, arXiv preprint 2312.10997, 2023.

[JD22] Jung, C.; Dörr, J.: Data Usage Control. In: Designing Data Spaces. Springer, pp. 129–146, 2022.

[Le20] Lewis, P. et al.: Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In. Curran Associates, Inc., pp. 9459–9474, 2020.

[Ot22] Otto, B.: The Evolution of Data Spaces. In (Otto, B.; ten Hompel, M.; Wrobel, S., eds.): Designing Data Spaces. Springer, pp. 3–15, 2022.

[PDR22] Pretzsch, S.; Drees, H.; Rittershaus, L.: Mobility Data Space. In (Otto, B.; ten Hompel, M.; Wrobel, S., eds.): Designing Data Spaces. Springer, pp. 343–361, 2022.

[PSB22] Pettenpohl, H.; Spiekermann, M.; Both, J. R.: International Data Spaces in a Nutshell. In (Otto, B.; ten Hompel, M.; Wrobel, S., eds.): Designing Data Spaces. Springer, pp. 29–40, 2022.

[To23] Touvron, H. et al.: Llama: Open and efficient foundation language models, arXiv preprint arXiv:2302.13971, 2023.