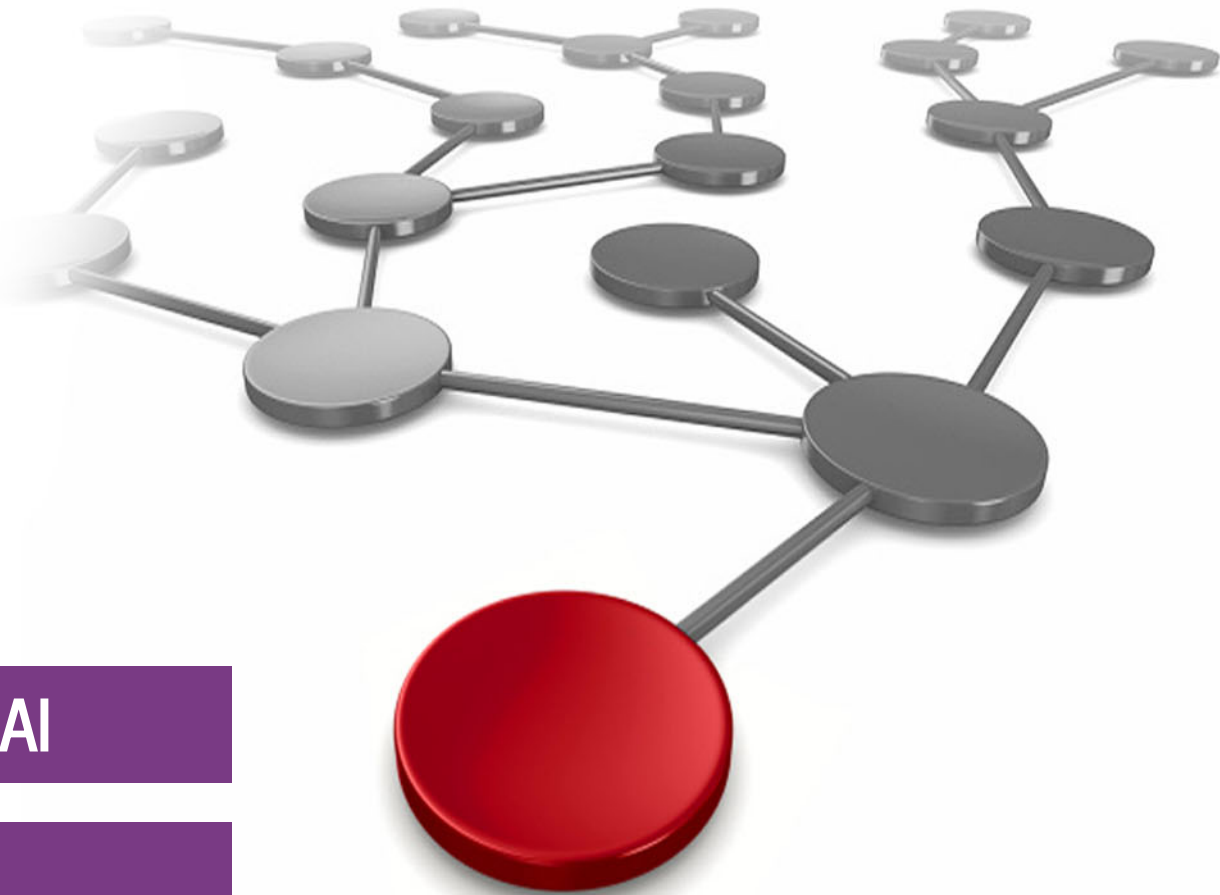


Next-generation Enterprise AI solutions with IBM watsonx.ai and IBM Storage Scale

Qais Noorshams
Chinmaya Mishra
Harald Seipp
Sailendu Patra
Dietmar Fischer
Kedar Karmarkar
Mathias Defiebre



Data and AI

Cloud



IBM Redbooks

January 2026

Note: Before using this information and the product it supports, read the information in “Notices” on page vii.

First Edition (January 2026)

This edition applies to Version 6, Release 0, Modification 0 of IBM Storage Scale System.

This document was created or updated on February 4, 2026.

© Copyright International Business Machines Corporation 2026. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices	v
Trademarks	vi
Preface	vii
Authors	vii
Now you can become a published author, too!	viii
Comments welcome	ix
Stay connected to IBM Redbooks	ix
Chapter 1. Introduction	1
1.1 The Evolution of Storage in The Age of Artificial Intelligence	2
1.2 Benefits of IBM Storage Scale Global Data Platform	4
1.3 IBM watsonx Ecosystem Overview	5
Chapter 2. Solution Architecture	7
2.1 Use Cases for IBM Storage Scale with IBM watsonx.ai	8
2.2 Solution Architecture	8
2.3 Architecture variations	9
2.4 Benefits of IBM Spectrum Scale for AI use cases	11
2.5 IBM Storage Scale and IBM watsonx.data	12
Chapter 3. Planning and Sizing	13
3.1 Hardware Requirements	14
3.1.1 Compute Cluster	14
3.1.2 Storage Cluster	14
3.2 Network Requirements	15
3.3 Software Requirements	15
3.4 Sizing Guidelines	16
3.4.1 IBM watsonx.ai	16
3.4.2 IBM watsonx.data	16
3.4.3 IBM Storage Scale System	17
Chapter 4. Configuring The Solution	19
4.1 Configuring IBM Storage Scale System and IBM Storage Scale	20
4.2 Configuring IBM Storage Scale CNSA	22
4.3 IBM Storage Scale CES S3	23
4.3.1 Install and Configure the IBM Storage Scale S3 Service	23
4.3.2 DNS load balancer for S3	24
4.3.3 Configuring Filesets as a Backing Storage for S3 buckets	25
4.3.4 Creating S3 buckets	26
4.4 Configuring Data Abstraction and Acceleration	28
4.5 Define IBM Storage Scale S3 buckets to IBM watsonx.data	30
4.6 Add a Milvus vector database service to IBM watsonx.data	32
Chapter 5. Examples, Use Cases and Solution application	37
5.1 Working With IBM watsonx.ai	38
5.1.1 GUI for an Interactive Workflow	38
5.1.2 Notebook for a programmatic workflow	39
5.2 Use Case: Robust Entity Extraction and Summarization With Docling	40

- 5.2.1 High-level Workflow Overview 41
- 5.2.2 Financial Summary Example 41
- 5.3 Use Case: Bulk data ingest and query evaluation with Open RAG Benchmark 45
 - 5.3.1 Data Ingest 45
 - 5.3.2 RAG Query 48
- 5.4 Use Case: RAG with Role-Based Content Filter on IBM Storage Scale 50
 - 5.4.1 Data Formats and Data Preparation 51
 - 5.4.2 Model Setup 52
 - 5.4.3 RAG Queries 53

- Related publications 59**
 - IBM Redbooks 59
 - Online resources 59
 - Help from IBM 60

- Index 61**

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing, IBM Corporation, North Castle Drive, MD-NC119, Armonk, NY 10504-1785, US

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at <https://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks or registered trademarks of International Business Machines Corporation, and might also be trademarks or registered trademarks in other countries.

DataStage®	IBM watsonx®	watsonx Code Assistant®
Granite®	Orchestrate®	watsonx Orchestrate®
IBM®	Redbooks®	watsonx.ai®
IBM Cloud®	Redbooks (logo)  ®	
IBM Spectrum®	watsonx®	

Evolution, are trademarks or registered trademarks of Kenexa, an IBM Company.

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

OpenShift, Red Hat, are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Preface

This IBM Redpaper describes the IBM solution for using IBM Storage Scale as enterprise storage with IBM watsonx®.ai. The paper showcases how IBM watsonx.ai® applications can benefit from the enterprise storage features and functions offered by IBM Storage Scale.

This Redpaper is targeted toward technical professionals (customers, consultants, technical support staff, IT Architects, and IT specialists) who are responsible for delivering AI-driven solutions. This Redpaper is relevant for:

- ▶ Technical professionals designing and implementing IBM watsonx.ai solutions
- ▶ Existing IBM Storage Scale customers looking to implement IBM watsonx.ai solutions

Authors

This paper was produced by a working at IBM Redbooks, Tucson Center.

Qais Noorshams is a Senior Software Engineer and Solution Architect within the IBM® Storage Scale organization. Since he joined IBM in 2015, he has been holding various technical and leadership positions in international software development projects. He is a certified Expert Developer, IBM Recognized Speaker, and IBM Recognized Teacher. His track record includes authoring more than 20 granted patents, more than 15 peer-reviewed publications, and various IBM-published newsletters and articles. He holds a diploma degree (Dipl.-Inform.) and a PhD degree (Dr.-Ing.) in Computer Science from Karlsruhe Institute of Technology (Germany).

Chinmaya Mishra is a Software Architect in the Big Data & Analytics team within the IBM Storage Scale organization in IBM ISDL Labs. He joined IBM India in 2001 and has hold various technical and leadership roles in software products and solutions development teams across IBM India and IBM US. His area of expertise includes Transaction processing, Operating systems, Cloud native solutions as a service, high performance clustered filesystems and Data & Analytics Solutions. He holds a Bachelor of Technology degree in Electrical Engineering from the Indian Institute of Technology, Kharagpur.

Harald Seipp is a Senior Technical Staff Member and Principal Solution Architect with IBM Client Engineering EMEA. He is responsible for creating complex Storage Cloud and Software Defined Storage (SDS) architectures and provide solutions to comply with challenging client requirements. As a worldwide leader in the area of Kubernetes, OpenShift and Object Storage solutions, he implements first-of-a-kind solutions in co-creation with clients during proof-of-experience pilot engagements. In prior job roles, he was founder and technical leader of the Cloud Storage Center of Excellence as part of the IBM Systems EMEA Storage Competence Center, lead architect for the primary IBM host-based Tape diagnostic tool, co-inventor of an IBM storage product, working on a cloud object storage research project and working as a software developer for more than 15 years at IBM and two other companies. He acts as a regular speaker at international technical conferences and holds various patents on storage and networking technology.

Sailendu Patra is a Solution Architect for Data & AI in IBM Client Engineering, based in Bangalore, India. He holds a Bachelor of Technology from BPUT, Odisha and a Master's

degree from IIT Kharagpur. With extensive experience in the industry, including a decade in data science, he specializes in architecting and delivering end-to-end AI and Generative AI solutions using IBM's Data & AI and watsonx portfolio, including watsonx.ai, watsonx.governance and watsonx.data. Sailendu has led numerous watsonx-based Generative AI engagements across industries and has been recognized in global AI competitions for his innovative contributions. An active contributor to the AI community, he has filed multiple patents and published research at the AAA-rated NeurIPS conference. He remains deeply involved in advancing innovation through hackathons, community initiatives and continuous exploration of emerging technologies.

Dietmar Fischer is the Manager of the IBM Storage Scale's AI, Big Data and Analytics team. Dietmar has been with IBM for more than 25 years and has held several positions within the IBM Storage development organization including software test, development, project management, and management. Dietmar has a strong technical computer science background and enjoys developing great solutions with a team of very talented experts.

Kedar Karmarkar is a Development Architect with the IBM Storage Scale development team and has contributed to Protocols, Data Caching, Scale containerization, AI solutions and Storage Scale Development adoption teams. Kedar has over 25 years of infrastructure software, storage development experience in management, and architect roles. Prior to IBM, Kedar has led development of network-attached storage (NAS), Block level virtualization and replication, systems, and storage management products. Kedar has a Bachelor of Engineering (Computer Science) degree from University of Pune, India

Mathias Defiebre is a leading IBM expert specializing in Data Analytics and Software Defined Storage (SDS) with over 25 years of experience in storage solutions. His current focus lies at the intersection of Machine Learning and SDS, aiming to optimize their integration. From IBM's EMEA Storage Client Engineering, he provides services to clients (Pilots and Deployments). He graduated from the University of Cooperative Education Mannheim with a German Diploma in Information Technology Management and a Bachelor of Science. Mathias also is a Master Certified IT Specialist. He is the author of several Storage Redbooks® publications.

Thanks to the following people for their contributions to this project:

Phillip Gerrard
IBM Redbooks, IBM Infrastructure

Ted Hoover
IBM Product Management

Christina Orosco
IBM Director

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an IBM Redbooks residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks

in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our papers to be as helpful as possible. Send us your comments about this paper or other IBM Redbooks publications in one of the following ways:

- ▶ Use the online **Contact us** review Redbooks form found at:

ibm.com/redbooks

- ▶ Send your comments in an email to:

redbooks@us.ibm.com

- ▶ Mail your comments to:

IBM Corporation, IBM Redbooks
Dept. HYTD Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- ▶ Find us on LinkedIn:

<https://www.linkedin.com/groups/2130806>

- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

<https://www.redbooks.ibm.com/subscribe>

- ▶ Stay current on recent Redbooks publications with RSS Feeds:

<https://www.redbooks.ibm.com/rss.html>



1

Introduction

Artificial Intelligence (AI) technologies are developing rapidly. The significant advancements in processing power achieved through the latest generations of graphics processing units (GPUs) enable the development and deployment of increasingly complex AI models. This leads to the creation and adoption of large-scale foundation models and large language models (LLMs). These models are the cornerstone for new emerging technologies like Retrieval Augmented Generation (RAG) or Agentic AI. At the core of all of these models is data. The ability to provide high quality data through a fast storage solution is key when committing costly resources and optimizing the IT infrastructure to develop new, high quality solutions. Often the ability to provide this data at speed is the difference between having to take a sub-par 'AI-like' approach and developing industry-leading AI-based solutions.

This IBM Redpaper describes IBM watsonx.ai as the premier AI platform using IBM Storage Scale as its backbone for a high-performance, highly reliable storage solution. The paper describes the architecture of the IBM watsonx.ai and IBM Storage Scale solution and its competitive advantages. Moreover, the paper explains planning and sizing guidelines as well as configuration details throughout the solution stack. Finally, practical applications and use cases show the approach in typical scenarios to showcase the solution architecture in action.

In this chapter:

- ▶ 1.1, “The Evolution of Storage in The Age of Artificial Intelligence” on page 2
- ▶ 1.2, “Benefits of IBM Storage Scale Global Data Platform” on page 4
- ▶ 1.3, “IBM watsonx Ecosystem Overview” on page 4

1.1 The Evolution of Storage in The Age of Artificial Intelligence

Artificial Intelligence (AI) is becoming the key ingredient across industries, promising a competitive advantage by unlocking hidden value in the vast amounts of proprietary customer data created and stored by modern companies. The rising adoption and capabilities of AI, however, lead to ever-increasing computing and storage requirements. At the same time, AI models require data to sustain them throughout their lifecycle, from training and testing to deployment and inference. The role and importance of storage is evolving in this new age of AI, see Figure 1-1

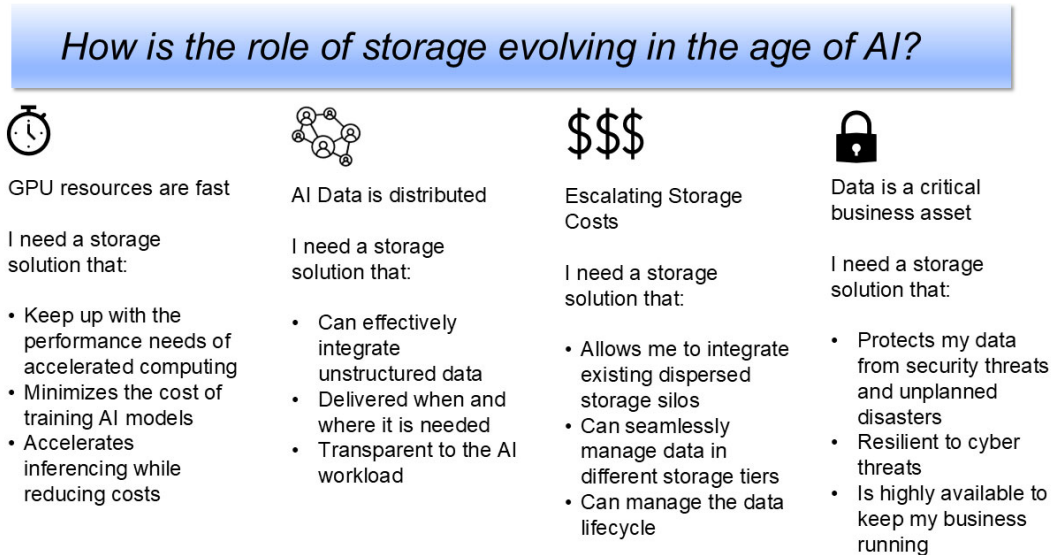


Figure 1-1 The role of storage in the age of AI

In summary, four challenge areas can be identified:

- ▶ Today's GPUs are fast, but they depend on storage solutions that can keep up with their performance needs. Slow storage leads to underutilized GPUs and a waste of resources. Fast storage minimizes the cost of training and accelerates inferencing times.
- ▶ Data needed for AI is barely ever fully structured and ready in one place. Effectively integrating distributed data, which includes vast amounts of unstructured data, is essential to ensure it is delivered at the right time and place where it is needed. Furthermore, this process needs to be transparent to the AI workload.
- ▶ Storage costs can suddenly explode if the data is excessively duplicated, moved, and copied. A well-designed storage solution needs to integrate well with distributed storage environments. It needs to effectively manage data across different cost-optimized storage tiers as well as manage the overall lifecycle of the data.
- ▶ Data at its core, is a critical business asset that needs to be protected from security threats and disasters. This includes processes for cyber resilience and disaster recovery that ensure the data is highly available to keep the business running.

Traditional storage environments typically struggle with at least one of these challenges. As illustrated in Figure 1-2 on page 3 the main AI phases that lead from data to decision often involve multiple data moves and copies. These main phases include:

- ▶ **Data collection:** Data gathered from organizational units within the company is stored in the corporate data lake, which is usually S3 or NFS connected.
- ▶ **Data preparation:** The data is loaded, processed, and filtered, e.g. for inappropriate data.

- ▶ **Data transfer:** To be able to process the data quickly, the data is moved into a high performance storage environment.
- ▶ **Training:** The data is loaded for training from high performance storage and accessed through POSIX or GPU Direct.
- ▶ **Evaluation and deployment:** Throughout the evaluation, frequent checkpoints are written back to storage to save intermediate results.

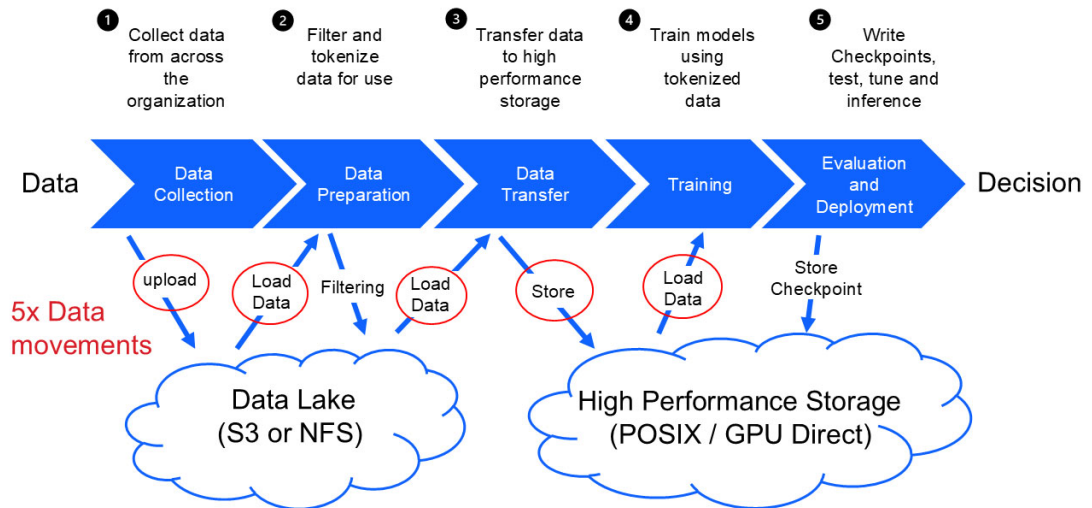


Figure 1-2 Problems of traditional storage approaches

This process no longer scales when the volume of data continue to grow. Hosting multiple data copies and managing many data movement processes is increasingly costly and tedious.

To address these challenges, IBM Storage and IBM Storage Scale offer a unique solution, the Global Data Platform, see Figure 1-3 on page 3. With the Global Data Platform, the data moves and copies are not required, and the Global Data Platform can provide heterogeneous access mechanisms to data sources like on-prem NAS Storages and Object stores, or remote Cloud Object Stores. The data is cached locally, backed by IBM Storage Scale System and tiered to tape when needed.

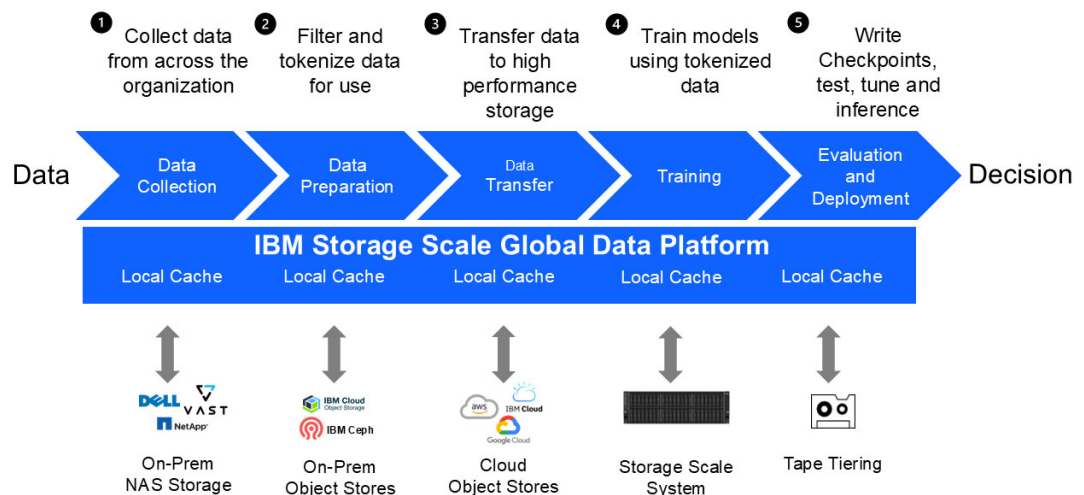


Figure 1-3 IBM Storage Scale Global Data Platform

1.2 Benefits of IBM Storage Scale Global Data Platform

The IBM Storage Scale Global Data Platform offers multiple advantages to address today's storage challenges. At its core is IBM Storage Scale, an enterprise-grade parallel file system that provides superior resiliency, scalability, and control. IBM Storage Scale delivers scalable capacity and performance to handle demanding data analytics, AI/ML, content repository and technical computing workloads.

The features of IBM Storage Scale include:

- ▶ Scalable performance and capacity
- ▶ Data caching and acceleration
- ▶ Multi-protocol support
- ▶ Update & deployment automation
- ▶ Data resiliency and privacy
- ▶ Intuitive management GUI, APIs, and CLI
- ▶ Container native support and CSI (Container Storage Interface)
- ▶ Quotas, QoS (Quality of Service), snapshots, tiering
- ▶ Audit logging and tracking
- ▶ and more...

IBM Storage Scale also incorporates technologies specifically developed for AI use cases, such as [Content-Aware Storage \(CAS\)](#). CAS leverages IBM Storage Scale as AI-optimized storage to build AI data pipelines using AI microservices, e.g. based on NVIDIA NIMs, specialized vector databases, and hardware accelerators, e.g. GPUs. IBM Storage Scale is also well-suited as a specialized cache for generative [AI use cases for so-called KV \(Key/Value\) caching](#). KV caching speeds up LLM inferencing by caching intermediate results thus improving both latency and throughput.

1.3 IBM watsonx Ecosystem Overview

IBM watsonx is integrated into a large ecosystem, a subset of which is introduced in the following. As illustrated in Figure 1-4, the deployment infrastructure builds the foundation and can be on-premises or in the cloud. Further, IBM watsonx is built atop Red Hat OpenShift as the hybrid cloud foundation and Red Hat OpenShift AI providing AI models and tools. The foundation also includes IBM Software Hub, which is a cloud-native solution to install, manage, and monitor IBM solutions on Red Hat OpenShift.

IBM watsonx is a portfolio of AI products and at its core includes IBM watsonx.ai, IBM watson.governance, and IBM watsonx.data. It is further extended with solutions like IBM watsonx Orchestrate® to build AI assistants and AI agents and IBM watsonx Code Assistant® for code generations and software development acceleration, for example. In addition to IBM watsonx, IBM Cloud® Pak for Data provides a modular set of integrated software components for data analysis, organization and management. On the other hand, IBM Data Product Hub is a solution built around data products and enables users to create, publish, and share data products between teams.



Figure 1-4 Solution ecosystem overview



Solution Architecture

A well-designed solution drives business value through efficient and effective operation, while the opposite will, most likely, introduce unforeseen challenges and costs. The high-level architecture planning is the foundation for a solution to effectively address the requirements.

This chapter begins with describing use cases for IBM Storage Scale and IBM watsonx.ai. Afterwards, this chapter shows the solution architecture as well as architecture variations to be able to adapt to specific requirements.

In this chapter:

- ▶ 2.1, “Use Cases for IBM Storage Scale with IBM watsonx.ai” on page 8
- ▶ 2.2, “Solution Architecture” on page 8
- ▶ 2.3, “Architecture variations” on page 9
- ▶ 2.4, “Benefits of IBM Spectrum Scale for AI use cases” on page 11
- ▶ 2.5, “IBM Storage Scale and IBM watsonx.data” on page 12

2.1 Use Cases for IBM Storage Scale with IBM watsonx.ai

The solution described in this section offers a state-of-the-art AI platform with distinct advantages and uses cases, which are described hereafter.

- ▶ **Modern AI infrastructure and platform:** IBM watsonx and IBM Storage Scale offer a modern AI infrastructure by design. Both inherently match well and are targeted to address enterprise requirements as well as their performance and scalability demands. They enhance enterprise hybrid cloud environments through their design, allowing to cache data and speedup data-intensive operations. IBM watsonx.ai is a platform that provides a suite of AI tools and capabilities, enabling users to build, train, and deploy custom AI models, access pre-built models, and integrate AI into their applications and workflows. IBM Storage Scale is a high-performance, highly scalable storage system with an elaborate ecosystem, tools, and features.
- ▶ **Disaggregated compute and storage infrastructure:** The design of IBM watsonx and IBM Storage Scale is based on the concept of disaggregated compute and storage infrastructure. Unlike monolithic designs, the flexible nature of IBM watsonx and IBM Storage Scale allow you to build, scale, and adapt compute and storage independently. This enables tailoring the environment to the use case and flexibly change it should the need arise.
- ▶ **Modular extension to existing IBM Storage Scale environments:** For customers with existing IBM Storage Scale environments, the solution provides a natural and modular extension on top of existing investments and without costly data copies and data movement into other data formats or data sources. IBM Storage Scale itself serves as the Data Lakehouse for the AI operations without requiring other third-party solutions.

2.2 Solution Architecture

The architecture of the solution as illustrated in Figure 2-1 on page 9, follows a layered design and is based on a compute infrastructure and a storage infrastructure. The compute infrastructure hosts Red Hat OpenShift running IBM watsonx.ai and IBM watsonx.data. IBM watsonx.ai manages and runs AI models, workflows, and applications and connects for data access to IBM watsonx.data, which acts as an abstraction layer to the storage infrastructure. IBM watsonx.data hosts vector databases, such as Milvus, backed by local buckets and provides abstraction in form of data formats, e.g., Apache Iceberg, and access to traditional databases. Furthermore, IBM watsonx.data supports data preparation using Apache Spark.

For scenarios requiring more specialized data transformation capabilities, IBM DataStage® can be deployed additionally. The storage backend is IBM Storage Scale and provides services for data access through multiple protocols, e.g., S3 and POSIX. IBM watsonx.data connects to object buckets exposed by IBM Storage Scale and accesses the IBM Storage Scale file system via the S3 protocol. In turn, IBM Storage Scale can act as an abstraction and caching tier for other cloud and on-premises data sources using a component that serves as storage cache for abstraction and acceleration, provided by the Active File Management (AFM). AFM connects to other data sources, pulling the data for caching as well as pushing new updates, which can speed up data processing significantly by unifying multiple distributed data sources. This setup also supports hybrid cloud deployment models and is particularly beneficial when the AI applications run on-premises and most of the data resides in the cloud. It significantly improves performance while reducing egress costs by caching data on IBM Storage Scale and exposing them as cached buckets to the application without having to repeatedly collect the data from the cloud.

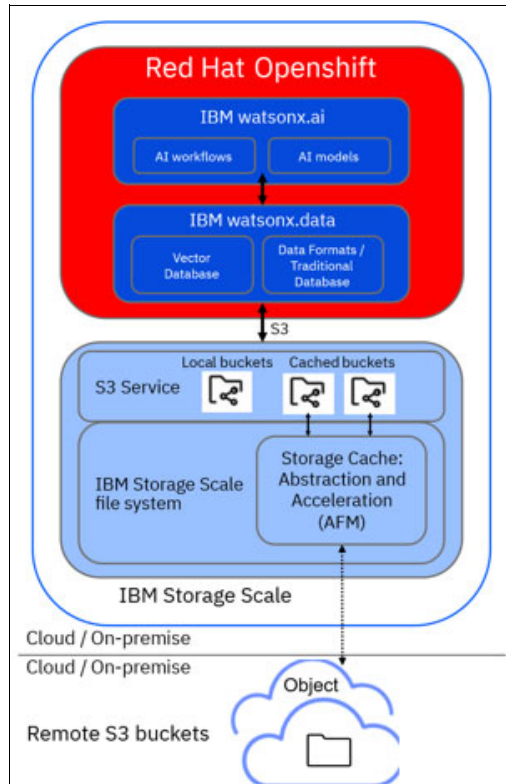


Figure 2-1 Solution architecture

2.3 Architecture variations

Depending on the use case, the solution architecture can be adapted. To this end, the following describes variations of the solution architecture. They primarily differ in the data access path. Furthermore, the flexibility in the compute infrastructure is illustrated in Figure 2-2 on page 10. For example, IBM watsonx.ai can be used for data preparation, hence accessing the IBM Storage Scale file system directly via S3 as well as through IBM watsonx.data. The AI models within watsonx.ai can be exploited for AI workflows. Moreover, the AI models are naturally employed for both training and inferencing scenarios. IBM watsonx.data as a higher-level layer provides abstractions in form of both data formats, e.g., Apache Iceberg, and data base connections, e.g., vector databases and relational databases.

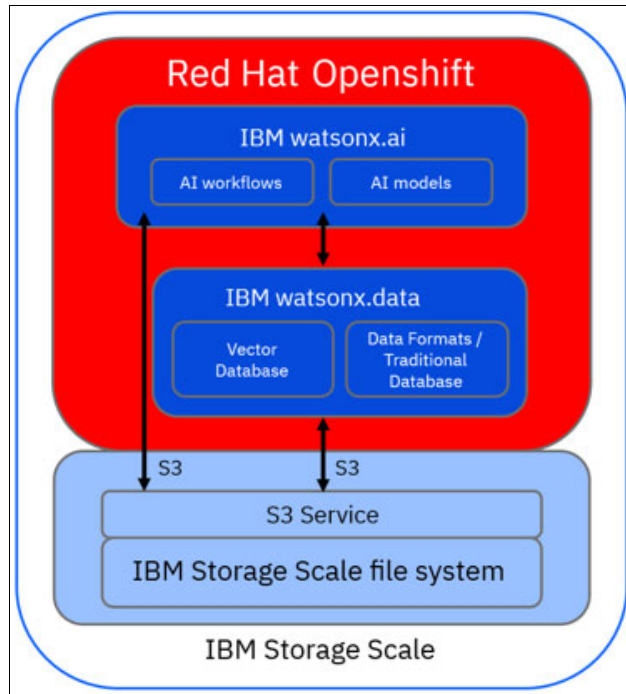


Figure 2-2 Solution architecture variation

If the abstractions IBM watsonx.data provides are not required, it can be omitted as described in Figure 2-2. IBM watsonx.ai can access data on IBM Storage Scale with the cloud-native integration IBM Storage Scale Container Native Storage Access (CNSA), which enables IBM Storage Scale as local storage and integrates the IBM Storage Scale file system via a remote mount. This helps for both data access as well as a cluster bring-up support when the required installation data can be kept on the shared storage file system and does not need to be copied.

There are several benefits to use IBM Storage Scale for configuring local storage for the cluster:

1. Customers don't have to invest in buying storage rich servers just to fulfill the local storage requirements for the installed services. The same centralized storage, IBM Storage Scale, can be used for application data as well as for local storage requirements.
2. Many generative AI models tend to have a large storage footprint and it can consume a lot of storage especially when the models are trained or fine-tuned repeatedly. Having IBM Storage Scale as a centralized shared storage is helpful in such cases.
3. Large models can load much faster when loading from a fast and parallel storage like Storage Scale.

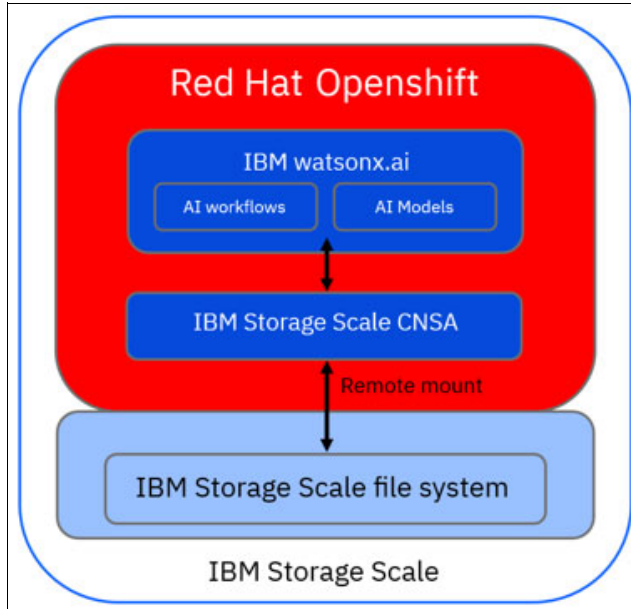


Figure 2-3 Solution architecture variation

Finally, accessing data directly on IBM Storage Scale using the S3 protocol is an option. Shown in Figure 2-4 on page 11, IBM watsonx.ai can provide AI workflows accessing data on IBM Spectrum® Scale via S3. Similarly, the data can be used as input for the AI models as well. This scenario is outlined here for the sake of completeness for use cases when IBM watsonx.ai can work on S3 data directly or when the abstractions provided by IBM watsonx.data are not required. For most use cases, the previous architectures are typically more applicable.

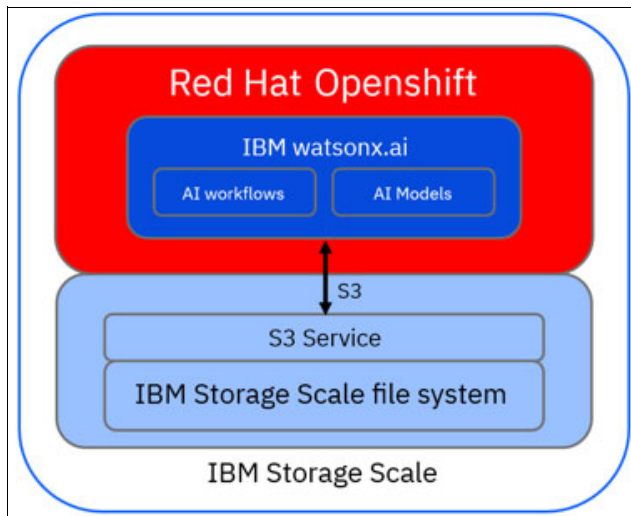


Figure 2-4 Solution architecture variation

2.4 Benefits of IBM Spectrum Scale for AI use cases

Why should a customer invest in IBM Storage Scale for their AI needs? The advantages are manifold throughout multiple aspects.

- ▶ **Economics:** IBM Spectrum Scale offers the highly competitive price performance. AI data centers host a large farm of expensive GPUs, leaving them idle or unutilized because the storage backend cannot provide the required data in time is a huge waste. IBM Spectrum Scale is designed for high-performance data access allowing to fully exploit and utilize the AI hardware in the data center.
- ▶ **Shared data access:** IBM Spectrum Scale offers simultaneous read and write access to the same data through multiple protocols, e.g., S3 and POSIX. Data can be accessed for its purpose and expensive data copy or data move processes can be avoided, which would lead to data silos that are expensive to manage and maintain.
- ▶ **Caching and tiering:** IBM Spectrum Scale allows data from remote sources to be cached with mechanisms like Active File Management (AFM). The data is held where it is needed and tiered on-demand to the most economically viable data tier, e.g., Flash vs. Tape.
- ▶ **Enterprise solution:** IBM Spectrum Scale has built-in enterprise ready concepts for data availability, reliability, and security. For instance, encryption both in flight and at rest are inherent security capabilities of IBM Spectrum Scale. Furthermore, fault-tolerant design, redundancy, and other concepts are employed to provide high-availability.

2.5 IBM Storage Scale and IBM watsonx.data

As described in this chapter, IBM Storage Scale is well integrated with IBM watsonx.data. While not a prerequisite for IBM watsonx.ai as shown in the solution architecture variations, IBM watsonx.data provides extended features and an abstraction layer on top of IBM Storage Scale. For example, it provides a vector database service for unstructured data search to support enhanced AI use cases. Selective features of IBM watsonx.data and configuration options for IBM Storage Scale are described in the remainder of this Redpaper. For more information specifically on IBM Storage Scale support for IBM watsonx.data, please refer to the authors' previous publication available at:

<https://www.redbooks.ibm.com/abstracts/redp5743.html>.



Planning and Sizing

Deployments in the field may start small as proof-of-concepts (POCs) and grow to meet the expected workload as the environment is phased into production. Moreover, production environments usually grow as well as the workload grows over time.

In this chapter, we discuss the hardware, network, and software requirements. We outline guidelines and estimates to plan the intended environment. Where applicable, we refer to product documentation for further information.

In this chapter:

- ▶ 3.1, “Hardware Requirements” on page 14
- ▶ 3.2, “Network Requirements” on page 15
- ▶ 3.3, “Software Requirements” on page 15
- ▶ 3.4, “Sizing Guidelines” on page 16

3.1 Hardware Requirements

For the overall solution, the hardware requirements can be grouped into two parts, the compute cluster and the storage cluster. In addition, we distinctly distinguish between POC / testing environments and production environments, because high availability (HA) may not be an issue for simple experiments and testing. On the other hand, production environments need redundant hardware, nodes and components to tolerate hardware failures.

3.1.1 Compute Cluster

For Red Hat OpenShift, experiments and initial testing can be done on as few as a single node including one GPU for the foundation models. A minimal HA environment consists of 3 control plane nodes and 2 worker nodes with one GPU each. The worker nodes can be scaled based on the workload requirements. Control plane nodes and worker nodes can be deployed on the same server, thus a minimal setup consists of 3 physical nodes.

- ▶ Control plane nodes: minimum 3 nodes, 4 CPUs, 16 GB RAM, 100 GB Storage
- ▶ Worker nodes: minimum 2 nodes, 2 CPUs, 8 GB RAM (in multiple cases, the foundation models require 96GB RAM or more, see below), 100 GB Storage, minimum one GPU with 48 GB RAM (100 GB RAM recommended)

For more details, you can refer to the Red Hat documentation at:

https://docs.redhat.com/en/documentation/openshift_container_platform/4.20/html/installing_on_bare_metal/user-provisioned-infrastructure.

The memory and storage as well as the GPU requirements for the IBM watsonx.ai foundation models are listed in the IBM documentation at:

<https://www.ibm.com/docs/en/software-hub/5.2.x?topic=install-foundation-models>.

3.1.2 Storage Cluster

The storage cluster is comprised of the following parts.

- ▶ The core of the storage cluster is IBM Storage Scale. It can be a software-defined storage cluster or based on the IBM Storage Scale System appliance. The IBM Storage Scale System is inherently redundant including two I/O nodes.
- ▶ For caching and acceleration, at least one node for an AFM gateway is required. It requires at least 128 GB RAM; the specific CPU and memory requirements depend on the number of files and filesets assigned to AFM. For HA, at least two nodes need to take on the AFM gateway role. More details can be found at:
<https://www.ibm.com/docs/en/storage-scale/6.0.0?topic=planning-afm>.
- ▶ To expose the IBM Storage Scale through the S3 protocol, a dedicated protocol cluster is employed. The S3 protocol is integrated into the Cluster Export Services (CES), which is used to enable other protocol access, e.g., via NFS, for IBM Storage Scale as well. The protocol cluster can be as little as a single node, but a three-node cluster is the minimal configuration for HA. In addition, the workload can be distributed across the protocol nodes for load balancing. More details on S3 support for IBM Storage Scale is available at:
<https://www.ibm.com/docs/en/storage-scale/6.0.0?topic=gpfs-s3-support-overview>.

3.2 Network Requirements

In the layered architecture, the network design spans two directions, intra-layer traffic among nodes of the compute and storage cluster, respectively, and inter-layer traffic between the compute and the storage cluster.

- ▶ **Intra-layer network:** The communication and network traffic within the compute cluster or the storage cluster is primarily sensitive to latency. This is because the infrastructure needs to keep itself consistent by checking states and updating metadata on changes. Since IBM Storage Scale is a distributed file system by design, this fact is especially true on the storage layer as any communication lags most likely immediately impact data throughput.
- ▶ **Inter-layer network:** The network connecting the compute and storage layers, which serves as the actual data path, is the primary focus for network bandwidth considerations. Starting from the source, data is being stored or cached from a remote location on the storage servers of the IBM Storage Scale file system. The data is passed through the protocol nodes to the compute cluster, where it is processed and the results possibly stored back through the protocol nodes on the storage server.

More details specifically on the network design and preparation for the IBM Storage Scale System can be found at:

<https://www.ibm.com/docs/en/storage-scale-system/storage-scale-software/6.2.3?topic=deployment-networking-preparation>.

For the compute layer, a minimum throughput rate of 350 MB/s between cluster nodes is required. Further requirements can we found at:

<https://www.ibm.com/docs/en/software-hub/5.2.x?topic=requirements-network>.

3.3 Software Requirements

The main software components and their requirements are the following:

- ▶ **IBM watsonx.ai:** There are two installation modes for IBM watsonx.ai, full service and lightweight engine. While the former installs all the services available in IBM watsonx.ai, the latter focuses on services for inferencing foundation models programmatically. Both installation flavors require Red Hat OpenShift Container Platform, Red Hat OpenShift AI, and IBM Software Hub. More information is available at:
<https://www.ibm.com/docs/en/software-hub/5.2.x?topic=install-choosing-installation-mode>.
- ▶ **IBM watsonx.data:** For IBM watsonx.data, the requirements are a subset of IBM watsonx.ai depending on Red Hat OpenShift Container Platform and IBM Software Hub. Both IBM waitsonx.ai and watson.data can be hosted within the same Red Hat OpenShift cluster. Installation details for IBM watsonx.data are described at:
<https://www.ibm.com/docs/en/software-hub/5.2.x?topic=watsonxdata-installing>.
- ▶ **IBM Storage Scale – S3 protocol support:** The S3 protocol is integrated into the Clustered Export Services (CES) and can be configured and controlled similar to other CES-managed protocols, e.g., NFS. The installation and setup of S3 support for IBM Storage Scale is described at:
<https://community.ibm.com/community/user/blogs/rajan-mishra1/2024/08/02/deploying>.
- ▶ **IBM Storage Scale – server cluster:** The main storage cluster is provided by IBM Storage Scale. It establishes a distributed, fault-tolerant file system with extended features for security, availability, and observability. Setting up and installing IBM Storage Scale is

described at:

<https://www.ibm.com/docs/en/storage-scale/6.0.0?topic=examples-installing-storage-scale-creating-cluster>.

- ▶ IBM Storage Scale Active File Management (AFM): The caching feature for external data sources is provided by AFM. It can pull data from on-premises storage platforms within the data center as well as remote storage systems like public Clouds. Installation considerations are outlined at <https://www.ibm.com/docs/en/storage-scale/6.0.0?topic=installing-installation-active-file-management-afm>.

3.4 Sizing Guidelines

For the components, this section shows recommended sizings. More details can be found in the product documentation.

3.4.1 IBM watsonx.ai

Recommended sizings for IBM watsonx.ai on x86 architecture are as follows. Individual services (e.g., Watson Studio) have varying resource requirements, which can be found together with an overview of the hardware requirements at:

<https://www.ibm.com/docs/en/software-hub/5.2.x?topic=requirements-hardware>.

Table 3-1 *watsonx.ai system requirements overview*

Node Role	Number of Servers	Number of vCPUs	Memory	Storage
Control Plane	3	4 vCPUs per node	16 GB RAM per node	N/A
Infra	3	4 vCPUs per node	24 GB RAM per node	N/A
Worker	3	16 vCPUs per node	128 GB RAM per node	500 GB per node
Load Balancer	2	2 vCPUs per node	8 GB RAM per node	100 GB

3.4.2 IBM watsonx.data

For IBM watsonx.data, the recommended sizings are given in the following. More details can be found at:

<https://www.ibm.com/docs/en/watsonxdata/standard/2.0.x?topic=requirements-hardware>

Table 3-2 *watsonx.data system requirements overview*

Node Role	Number of Servers	Number of vCPUs	Memory	Storage
Master + Infra	3 (both roles on the same node)	8 vCPU per node	32 GB RAM per node	N/A

Node Role	Number of Servers	Number of vCPUs	Memory	Storage
Worker	3	16 vCPU per node	128 GB RAM per node	300 GB per node
Load Balancer	2	4 vCPU per node	8 GB RAM per node	100 GB

3.4.3 IBM Storage Scale System

Size and specification of IBM Storage Scale System models are given in the following. The full data sheet can be seen at:

<https://www.ibm.com/downloads/documents/us-en/10a99803f5afd8d6>.

Table 3-3 IBM Storage Scale Requirements overview

	IBM Storage Scale 3500	IBM Storage Scale 6000
Size	2 rack units (RU)	4 rack units (RU)
Maximum Capacity	24 x 30.72TB NVMe	48 x 30.72TB NVMe 48 x 38TB FCM4 modules
Maximum Throughput	126 GB/s	330 GB/s
Expansion	Up to 4 direct-attached JBODs	Up to 9 direct-attached JBODs
Data Transfer	12Gb SAS	24 Gb SAS



4

Configuring The Solution

When deploying the proposed solution, there are multiple components and applications that require setup. The configurations also depend on the use case, for example, where an object bucket on the storage layer is defined locally or needs to be a locally cached, accelerated bucket from a remote source.

In this chapter, we show the storage configurations required for the solution architecture. This spans, among others, configuring IBM Storage Scale and the file system, how S3 buckets are exposed for the applications, and how IBM watsonx accesses the data on the storage layer.

In this chapter:

- ▶ 4.1, “Configuring IBM Storage Scale System and IBM Storage Scale” on page 20
- ▶ 4.2, “Configuring IBM Storage Scale CNSA” on page 22
- ▶ 4.3, “IBM Storage Scale CES S3” on page 23
- ▶ 4.4, “Configuring Data Abstraction and Acceleration” on page 28
- ▶ 4.5, “Define IBM Storage Scale S3 buckets to IBM watsonx.data” on page 30
- ▶ 4.6, “Add a Milvus vector database service to IBM watsonx.data” on page 32

4.1 Configuring IBM Storage Scale System and IBM Storage Scale

An example deployment following the recommended architecture as outlined in Figure 4-1. The deployment follows a tiered architecture and specifically the separation of compute and storage infrastructure. The three tiers are further identified as follows:

- ▶ The compute tier contains the application layer. It is running the Red Hat OpenShift Container Platform, which hosts IBM watsonx.ai and IBM watsonx.data. Their services use the S3 object protocol to access the storage infrastructure. Alternatively, or in addition, IBM Storage Scale CNSA can be employed as well to remotely mount the IBM Storage Scale file system of the IBM Storage Scale server cluster.
- ▶ The IBM Storage Scale client cluster contains the S3 protocol nodes for IBM watsonx S3 access. The IBM Storage Scale client cluster shown contains multiple Protocol nodes for high-availability. An uneven number of nodes is recommended to achieve quorum, but an even number with Tiebreaker configuration can be used as well. The nodes of the IBM Storage Scale client cluster remotely mount the file system(s) of the IBM Storage Scale server cluster.
- ▶ The IBM Storage Scale server cluster consists of IO server nodes and another node acting as AFM gateway. The Scale server cluster owns the file system and acts as an acceleration layer using AFM to connect with other S3, NFS data sources, or a remote IBM Storage Scale file system.

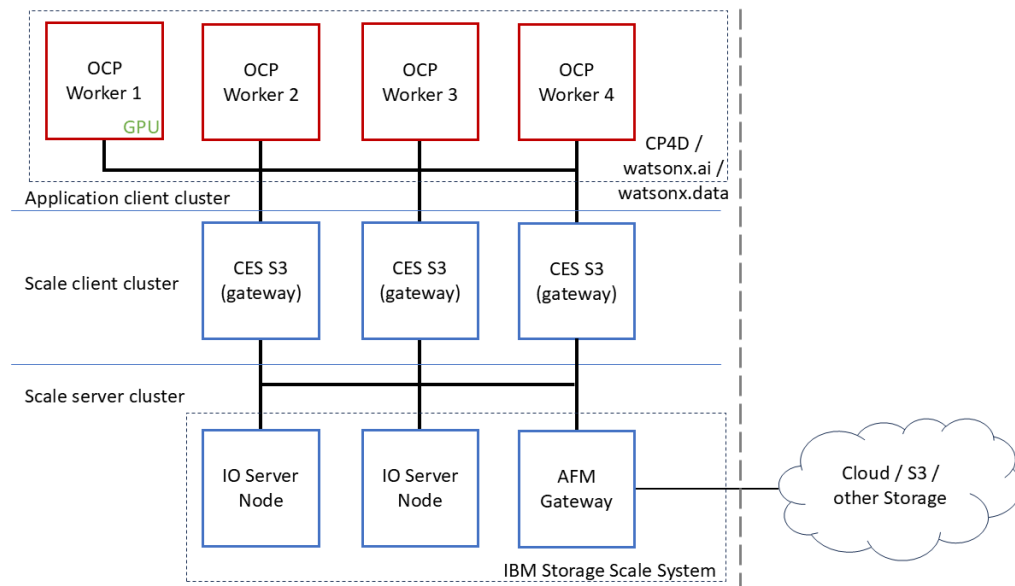


Figure 4-1 Three tiered deployment for IBM watsonx.ai and IBM watsonx.data with IBM Storage Scale

Example 4-1 on page 20 shows the configurations of the IBM Storage Scale client cluster used as example for this paper.

Example 4-1 Configurations of the IBM Storage Scale client cluster used in this Redpaper

```
[root@fscs-sr650-46 ~]# mmisccluster
```

```
GPFS cluster information
```

```
=====
```

```
GPFS cluster name:cess3gpfs.bda.scale.ibm.com GPFS cluster id:7776919622712034828
```

GPFS UID domain:cess3gpfs.bda.scale.ibm.com Remote shell command:/usr/bin/ssh
Remote file copy command: /usr/bin/scp Repository type:CCR

Node Designation	Daemon node name	IP address	Admin node name
1 quorum-manager-perfmon	node46s.bda.scale.ibm.com	10.10.1.86	node46s.bda.scale.ibm.com
2 quorum-manager-perfmon	node47s.bda.scale.ibm.com	10.10.1.87	node47s.bda.scale.ibm.com
3 quorum-manager-perfmon	node48s.bda.scale.ibm.com	10.10.1.88	node48s.bda.scale.ibm.com

The CES IPs are configured and assigned to the nodes of the IBM Storage Scale client cluster, in Example 4-2, we have two CES IPs setup.

Example 4-2 Listing the CES IP Addresses

```
[root@fscs-sr650-46 ~]# mmces address list --by-node
Node Daemon node name          IP address  CES IP address list
-----
1   node46s.bda.scale.ibm.com 10.10.1.86
2   node47s.bda.scale.ibm.com 10.10.1.87  10.10.1.121
3   node48s.bda.scale.ibm.com 10.10.1.88  10.10.1.120
```

Example 4-3 shows the IBM Storage Scale client cluster has mounted two file systems, one for data access, the other acting as CES shared root.

Example 4-3 Listing the remote mounted filesystems in the IBM Storage Scale client cluster

```
[root@fscs-sr650-46 ~]# mmremotefs show all
Local Name Remote Name Cluster nameMount PointMount OptionsAutomount Drive
Priority
essDataessDataess3k5.bda.scale.com /gpfs/essDataerwno-0
essCesRoot
essCesRootess3k5.bda.scale.com /gpfs/essCesRootrwno-0
```

The IBM Storage Scale server cluster owns the file systems and is described is described in Example 4-4.

Example 4-4 Configurations of the IBM Storage Scale server cluster used in this Redpaper

```
[root@fscs-sr650-36 ~]# mmisccluster

GPFS cluster information
=====
GPFS cluster name:ess3k5.bda.scale.com GPFS cluster id:213597018859291206
GPFS UID domain:ess3k5.bda.scale.com Remote shell command:/usr/bin/ssh
Remote file copy command: /usr/bin/scp Repository type:CCR

Node Daemon node name          IP address  Admin node name
Designation
-----
1   ess3k5a.bda.scale.ibm.com 10.10.1.185  ess3k5a.bda.scale.ibm.com
quorum-manager-perfmon
```

```

2   ess3k5b.bda.scale.ibm.com 10.10.1.186 ess3k5b.bda.scale.ibm.com
quorum-manager-perfmon
3   ems3k5.bda.scale.ibm.com10.10.1.76      ems3k5.bda.scale.ibm.com
quorum-manager-gateway-perfmon

```

The IBM Storage Scale client cluster, containing the S3 service, is used to configure S3 access for IBM watsonx. This process is explained more elaborately over the next sections. First, an S3 account is needed, which can be created after the corresponding user and group have been defined within the operating system. For this account, a S3 bucket is created afterwards. The combination of CES IP and port, account credentials (access key and secret key), and bucket name are required to define the connection from IBM watsonx.

4.2 Configuring IBM Storage Scale CNSA

The IBM Storage Scale CNSA cluster can be used to access the IBM Storage Scale file system. IBM Storage Scale CNSA is configured on the same Red Hat Openshift container cluster running the IBM watsonx services, by following the IBM documentation available at: <https://www.ibm.com/docs/en/scalecontainernative/6.0.0>.

The high-level steps include the following:

- ▶ Preparing the physical IBM Storage Scale cluster for container access. This includes configuring filesystem(s) for remote mount in the CNSA cluster, as well as creating the users and groups for the container operator and CSI users
- ▶ Configuring the worker and master nodes on the Red Hat Openshift cluster to prepare for the CNSA cluster setup.
- ▶ Deploying the CNSA cluster and the remote mount filesystem
- ▶ The CNSA cluster remotely mounts the file system(s) of the IBM Storage Scale server cluster and exposes the storage through Kubernetes Custom Resources called Persistent Volume (PV). The PVs are consumed by Red Hat Open Shift applications through Persistent Volume Claims (PVC) to satisfy their persistent storage requirements.

The following Example 4-5 shows the CNSA cluster CR artifact and the pods that belong to the cluster:

Example 4-5 Configurations of the Scale CNSA cluster used in this Redpaper

```

$ oc get clusters.scale.spectrum.ibm.com
NAME                EDITION    AGE
ibm-spectrum-scale  data-access 230d

[root@fscs-sr650-36 ~]# $ oc get pods -n ibm-spectrum-scale
NAME                                READY   STATUS    RESTARTS   AGE
compute-x1                          2/2     Running   0           109d
compute-x2                          2/2     Running   0           109d
compute-x3                          2/2     Running   0           109d
compute-x4                          2/2     Running   0           109d
ibm-spectrum-scale-gui-0             4/4     Running   0           109d
ibm-spectrum-scale-gui-1             4/4     Running   0           109d
ibm-spectrum-scale-pmcollector-0     2/2     Running   0           109d
ibm-spectrum-scale-pmcollector-1     2/2     Running   0           109d

$ oc get filesystem -n ibm-spectrum-scale -o wide

```

NAME	ESTABLISHED	REMOTE CLUSTER	MAINTENANCE MODE	AGE
remote-sample	True	remotecluster-sample	not supported	229d

The following Example 4-6 shows the storage classes. The storage class *ibm-spectrum-scale-internal* is an internal storage class exposed by the CNSA cluster and is internally used by the GUI and pmcollector services. The other storage class *ibm-spectrum-scale-sample* is used by watsonx services and applications.

Example 4-6 Storage classes exposed by the IB Storage Scale CNSA cluster used in this Redpaper

```
$ oc get storageclass |grep scale
ibm-spectrum-scale-internal          kubernetes.io/no-provisioner
Delete                               WaitForFirstConsumer             false                             230d
ibm-spectrum-scale-sample           spectrumscale.csi.ibm.com
Delete                               Immediate                         true                             229d
```

4.3 IBM Storage Scale CES S3

This section describes the steps to install and configure the IBM Storage Scale S3 service, followed by how to create S3 buckets.

4.3.1 Install and Configure the IBM Storage Scale S3 Service

To setup and install IBM Storage Scale, a convenient method is to use the Installation Toolkit. For general steps to install IBM Storage Scale, refer to <https://www.ibm.com/docs/en/storage-scale/6.0.0?topic=installing>. The following highlights the steps to install and enable the S3 service:

- ▶ Setup the basic information of the Scale cluster using the Installation Toolkit. Follow the documentation provided in the link above.
- ▶ After the basic information is setup, define protocol nodes for the cluster using:

```
# cd /usr/lpp/mmfs/6.0.0.0/ansible-toolkit/
# ./spectrumscale node add hostname -p ...
```

- ▶ Define the CES IPs for the cluster to be exported using:

```
# ./spectrumscale config protocols -e <EXPORT_IP_POOL>
```

Where <EXPORT_IP_POOL> is a comma separated list of IP Addresses that would be used by applications to access the S3 service.

Note: Reverse DNS lookup needs to be available for all CES IPs. The CES IPs must be unique and cannot be cluster node IPs.

- ▶ Configure the CES shared root file system, which is used for configuration and administration of the CES protocols, using:

```
# ./spectrumscale config protocols -f essCesRoot-m /gpfs/essCesRoot
```

Note: It is recommended that the CES shared root is a separate file system. The CES shared root needs to be at least 4 GB.

- ▶ Enable the S3 protocol using:

```
# ./spectrumscale enable s3
```

- ▶ Review the configuration and perform a precheck of the deployment using

```
# ./spectrumscale node list
# ./spectrumscale deploy -precheck
```

- ▶ Finally, deploy the changes using:

```
# ./spectrumscale deploy
```

- ▶ Verify that the S3 services are up and running on the designated S3 protocol nodes:

```
# mmces service list -a
node46s.bda.scale.ibm.com: S3 is running
node47s.bda.scale.ibm.com: S3 is running
node48s.bda.scale.ibm.com: S3 is running
```

- ▶ To view the S3 protocol configuration, run

```
# mms3 config list
```

Example 4-7 Default configuration of the S3 service

```
# mms3 config list S3 Configuration:
=====
ALLOW_HTTP : false
DEBUGLEVEL : default
ENABLEMD5 : false
ENDPOINT_FORKS : 2
ENDPOINT_PORT : 6001
ENDPOINT_SSL_PORT : 6443
GPFSDLPATH : /usr/lpp/mmfs/lib/libgpfs.so
NC_MASTER_KEYS_GET_EXECUTABLE : /usr/lpp/mmfs/bin/cess3_key_get
NC_MASTER_KEYS_PUT_EXECUTABLE : /usr/lpp/mmfs/bin/cess3_key_put
NC_MASTER_KEYS_STORE_TYPE : executable
NSFS_DIR_CACHE_MAX_DIR_SIZE : 536870912
NSFS_DIR_CACHE_MAX_TOTAL_SIZE : 1073741824
NSFS_NC_CONFIG_DIR_BACKEND : GPFS NSFS_NC_STORAGE_BACKEND : GPFS UVTHREADPOOLSIZE
: 16
```

4.3.2 DNS load balancer for S3

For increased performance, we configured a load balancer for the IBM Storage Scale S3 service. A simple Domain Name System (DNS) based load-balancer was chosen. A DNS entry is created with all the S3 Endpoint IPs as A records. Clients resolve the DNS entry to one random IP from the list, and cache it for TTL (time-to-live) duration.

DNS load balancers usually provide the best performance with no extra network hop, unlike other options such as Network Load Balancer (NLB) or Application Load Balancer (ALB). It is trivial to maintain and no extra hardware is needed. However, the downside is that balancing is random and does not consider host contention, which can lead to sub-optimal workload distribution.

Our DNS Loadbalancer (dnsmasq based) was created with the following configuration by adding a DNS A record for the 3 CES IPS, followed by restarting the dnsmasq service.

```
10.10.1.119 cesip.bda.scale.ibm.com
10.10.1.120 cesip.bda.scale.ibm.com
10.10.1.121 cesip.bda.scale.ibm.com
```

The DNS name cesip.bda.scale.ibm.com is then used for configuring IBM watsonx services to reach the IBM Storage Scale S3 service.

4.3.3 Configuring Filesets as a Backing Storage for S3 buckets

A fileset is a subtree of an IBM Storage Scale file system that in many respects appears like an independent file system. Filesets provide a means of partitioning the file system to allow administrative operations at a finer granularity than the entire file system:

Filesets can have their own defined quotas for data and inodes. The owning fileset becomes an attribute of each file for enforcing IBM Storage Scale based policies (such as automated tiering and placement, encryption, compression) as needed. Each fileset mounts at a regular directory path (called JunctionPath) within the Scale file system. A regular S3 bucket may be defined over the mount path.

For more information, see:

<https://www.ibm.com/docs/en/storage-scale/6.0.0?topic=scale-filesets>.

Here are some scenarios, where it may be preferred to create a S3 bucket over a fileset rather than over a regular directory:

- ▶ Assign a quota to the storage space that a bucket may consume
- ▶ Limit the total number of objects that a bucket may consume
- ▶ Automatically encrypt all of the bucket's data (objects) on the disk.
- ▶ Define automating tiering and placement policies for a bucket. For example, if cold data is uploaded to a bucket, automatically move it to a slower storage tier.
- ▶ Create time-based snapshots of the storage at a bucket level. Fileset snapshots can be created instead of creating a snapshot of an entire file system.

Example 4-8 shows how to create an independent fileset and create a S3 bucket on top of it.

Example 4-8 Creating an independent fileset

```
# mmcrfileset essData fset-1 --inode-space
new Fileset fset-1 created with id 1 root inode 524291.
```

Link the fileset to an IBM Storage Scale mount path.

```
# mmlinkfileset essData fset-1 -J /gpfs/essData/watsonx/b-wxd-fset1
Fileset fset-1 linked at /gpfs/essData/watsonx/b-wxd-fset1
```

Then proceed to create a S3 bucket over the fileset's mount path (directory), described in the following section. In the above example, the mount path is `/gpfs/essData/watsonx/` which would be the default bucket path (`--newBucketsPath`) for our S3 account.

4.3.4 Creating S3 buckets

This section explains how to configure S3 buckets over IBM Storage Scale. For every new bucket, a new directory is created under the Storage Scale file system.

Alternatively, a bucket could be configured over a pre-existing directory. For example, a bucket could be configured over the mount point directory of a Storage Scale fileset, so that the fileset becomes the backing storage of the S3 bucket.

The steps to create a S3 bucket are shown in the following command listings.

- ▶ First create a S3 account, associating the account to a system user.

```
# /usr/lpp/mmfs/bin/mms3 account create <S3 account-name> --uid <uid> --gid <gid>
--newBucketsPath <Path>
```

Where,

<uid> and <gid> are the Posix UID and GID associated with the S3 account. These parameters are not needed to be passed if a account name is passed. The account name should be a valid system username.

<Path> is filesystem absolute path, which will act as a base path for S3 buckets created using S3 API. This path can be overridden for buckets created with the `mms3 bucket create` command.

To view the details associated with this S3 account including its AWS access credentials, run

```
# mms3 account list <S3 account-name>
```

Then, create one or more S3 buckets, corresponding to this S3 account.

There are two ways a bucket can be created:

- ▶ **Method 1 - Using the `mms3` command:**

```
# mms3 bucket create <S3 bucket-name> --accountName <S3 account-name>
--filesystemPath <Path>
```

Where

<S3 account-name> is the name of account which should be used for the bucket

<Path> is the filesystem absolute path including the directory for the bucket, which is to be used for bucket creation. This could be different than the default bucket path (`--newBucketsPath`) configured for the S3 account.

The command will create a new directory with system path `<Path>` which will correspond to the S3 bucket.

Example 4-9 Steps to create a new S3 bucket with `mms3`

1. Creating a S3 account named "watsonx"

```
# mms3 account create watsonx --uid 2002 --gid 100 --newBucketsPath
/gpfs/essData/watsonx/
```

2. View the details associated with this S3 account including its AWS access credentials

```
# mms3 account list watsonx
NameNew Buckets PathUid  Gid Access KeySecret Key

watsonx /gpfs/ess3k54/watsonx/ 2002 100 <Our AWS_ACCESS_KEY_ID>< Our
AWS_SECRET_ACCESS_KEY>
```

3. Creating a S3 bucket named under the default Bucket path

```
# mms3 bucket create b-watsonx --accountName watsonx --filesystemPath
/gpfs/essData/watsonx/b-watsonx
```

4. Creating a S3 bucket named not under the default Bucket path

```
# mms3 create b-watsonx2 --accountName watsonx --filesystemPath
/gpfs/essData/b-watsonx2
```

5. Creating a bucket over an existing directory, containing data in it, in the following way:

- ▶ Change ownership of that directory to that of the S3 account first. For example.

```
# chown watsonx:users /gpfs/essData/data-dir1
```

- ▶ For Fileset backed storage, the mount paths are created as owned by root. Change ownership of that directory to that of the S3 account.

```
# chown watsonx:users /gpfs/essData/watsonx/b-wxd-fset1
# mms3 bucket create b-watsonx3 -accountName watsonx -filesystemPath
/gpfs/essData/data-dir1
```

Note: The directory '/gpfs/essData/data-dir1' for bucket already exists. Skipping update of ownership and the setting of permissions of the directory for the user with uid:gid=2002:100

Bucket b-watsonx3 created successfully

▶ **Method 2 - Using the S3 API, e.g. the “aws” S3 client as shown in Example 4-10 on page 27.**

Example 4-10 Creating an S3 bucket using the S3 API

```
# wget https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip # unzip
awscli-exe-linux-x86_64.zip

# cd aws
# ./install
# alias s3u2='AWS_ACCESS_KEY_ID=<Your AWS_ACCESS_KEY_ID>
AWS_SECRET_ACCESS_KEY=<Your AWS_SECRET_ACCESS_KEY> aws --endpoint
https://10.11.94.182:6443 s3'

# s3u2 mb s3:// b-watsonx2 make_bucket: b-watsonx2
```

4.4 Configuring Data Abstraction and Acceleration

Configure IBM Storage Scale AFM to enable Storage Abstraction and Acceleration for dispersed buckets. This workflow involves:

- ▶ Creating a rule in AFM defining the connection to the remote S3 bucket. This will create a fileset in IBM Storage Scale corresponding to the remote S3 bucket.
- ▶ Creating a S3 bucket over that fileset. This bucket thus created abstracts or virtualizes the remote object bucket and is exposed through the IBM Storage Scale S3 interface.

Follow these instructions for configuring storage acceleration over remote buckets:

To start with, designate one or more nodes as AFM nodes. To designate a node as a AFM node, first ensure that the node has the AFM rpm (`gpfs.afm.cos.*`) installed, and the node has necessary connectivity to the remote cloud object S3 endpoint. Then run:

```
# mmchnode --gateway -N <AFM node hostname>
```

- ▶ Get the AWS access key ID and secret key for your remote bucket instance. For example, if using IBM Cloud Object Storage (COS) as service on IBM Cloud, navigate to `cloud.ibm.com` Instances Storage → Service Credentials Tab expand on down arrow. Get the details from `cos_hmac_keys`.

- ▶ Login to an AFM gateway node.

- ▶ Create the access keys in AFM corresponding to the remote object bucket.

```
# mmafmcoskeys bucket[:{[Region@]Server|ExportMap}] set {<access key> <secret key> | --keyfile filePath}
```

- ▶ Create an AFM relationship for the remote S3 bucket as shown in Example 4-11.

Example 4-11 Creating AFM relationship for a remote S3 bucket

```
# mmafmcosconfig <Device> <FilesetName> --endpoint
http[s]://{[Region@]Server|ExportMap}[:port]--object-fs--bucket
<BucketName>--mode <AccessMode> --dir <Path> --debug
```

Where,

<Device> is the name of your Storage Scale filesystem

<FilesetName> is name of the fileset that you want created corresponding to the remote S3 object

<BucketName> is the name of the actual remote S3 bucket

<mode> is the AFM Access Mode

<Path> is the relative directory path under the filesystem mount directory, where you want the fileset to be mounted (LinkPath).

Note: The `--dir` parameter passed to the command. This is done to ensure that the fileset is created under the S3 “New Buckets Path” (from the command “`mms3 account list`”).

- ▶ To see the newly created fileset, run:

```
# mmlsfileset <Device>
```

To see the relationship of the fileset with the remote bucket, run:

```
# mmafmctl <Device> getstate
```

Where <Device> is the name of the Storage Scale filesystem

- ▶ Create a S3 bucket over the fileset's mount path. Change the ownership of the directory to that of the account corresponding to the S3 bucket.

```
# chown<s3 account user>:<s3 account group> <fileset mount path>
```

Create the S3 bucket pointing to that directory:

```
# mms3 bucket create <bucket-name>--accountName <S3 account name>
--filesystemPath <fileset mount path>/<bucket-name>
```

Example:

In this example, there is a remote S3 bucket named “chm-cos-s3-bucket” residing on IBM Cloud Object Storage (IBM COS). The following steps illustrate the steps for creating a virtual/accelerated S3 bucket named “b-watsonx” corresponding to the IBM COS bucket.

- ▶ Designate an AFM node:

```
# mmchnode --gateway -N ess3200b.bda.scale.ibm.com
```

- ▶ Run the following commands on the AFM node:

- Create access keys in AFM corresponding to the IBM COS object:

```
# mmafmcoskeys
```

```
chm-cos-s3-bucket:s3.us-east.cloud-object-storage.appdomain.cloud \ set
<Your AWS_ACCESS_KEY_ID> \ <Your AWS_SECRET_ACCESS_KEY>
```

View the AFM access key

```
# mmafmcoskeys all get --report version=1
```

```
chm-cos-s3-bucket:s3.us-east.cloud-object-storage.appdomain.cloud=COS:<Your
AWS_ACCESS_KEY_ID>:<Your AWS_SECRET_ACCESS_KEY>
```

- ▶ Create an AFM relationship for the IBM COS bucket:

```
# mmafmcosconfig essData ibmcos-bucket \
--endpoint http://s3.us-east.cloud-object-storage.appdomain.cloud \
--object-fs \
--bucket chm-cos-s3-bucket \
--mode iw \
--dir watsonx/ibmcos-bucket --debug
```

Note: the value of `-dir` parameter above is chosen so, because the filesystem `essData` mount path is `/gpfs/essData` and the default path for S3 buckets (from the command “`mms3 account list`”) is `/gpfs/essData/watsonx`. This has the effect of creating the mount point of the fileset `ibmcos-bucket` at a relative path `watsonx/ibmcos-bucket` corresponding to the filesystem mount point, over which we will create a S3 bucket later.

The command produces following output shown in Example 4-12.

Example 4-12 Output of the `mmafmcosconfig` command

```
# mmafmcosconfig essData ibmcos-bucket \
--endpoint http://s3.us-east.cloud-object-storage.appdomain.cloud \
```

```
--object-fs \
--bucket chm-cos-s3-bucket \
--mode iw \
--dir watsonx/ibmcos-bucket --debug

afmobjfs=essData fileset=ibmcos-bucket
bucket=chm-cos-s3-bucket newbucket= objectfs=yes dir=watsonx/ibmcos-bucket policy=
tmpdir= tmpfile= noDirectoyObject=no mode=iw remoteUpdate=no
xattr=no ssl=no autoRemove=no fastReaddir=no acl=no gcs=no vhb= cleanup=no
fastReaddir2=no lazyMigrate=no azure=no
bucketName=chm-cos-s3-bucket region=
serverName=s3.us-east.cloud-object-storage.appdomain.cloud cacheFsType=http map=
cacheHost=s3.us-east.cloud-object-storage.appdomain.cloud
Linkpath=/gpfs/essData/watsonx/ibmcos-bucket
target=http://s3.us-east.cloud-object-storage.appdomain.cloud/chm-cos-s3-bucket
endpoint=s3.us-east.cloud-object-storage.appdomain.cloud ENDPOINT=--endpoint
http://s3.us-east.cloud-object-storage.appdomain.cloud
XOPT= -p afmParallelWriteChunkSize=0 -p afmParallelReadChunkSize=0
```

- To view the newly created AFM relationship, run the `mmafmctl` command:

Example 4-13 mmafmctl output

```
# mmafmctl essData getstate
Fileset Name
Fileset Target
Gateway Node Queue Length Queue numExec Cache State

ibmcos-bucket
http://s3.us-east.cloud-object-storage.appdomain.cloud:80/chm-cos-s3-bucket Active
ems3000.bda.scale.ibm.com 0 3
```

Note: The output shows the AFM gateway node(s). The Cache State should be “Active” to indicate that the storage acceleration is working properly.

- Now create a S3 bucket over the directory `/gpfs/essData/watsonx/ibmcos-bucket`

Example 4-14 S3 bucket creation

```
# chown watsonx:users ibmcos-bucket/

# mms3 bucket create b-watsonx-cos --accountName watsonx --filesystemPath
/gpfs/essData/watsonx/ibmcos-bucket
Starting to create bucket with name b-watsonx-cos
```

Note: The directory `'/gpfs/essData/watsonx/ibmcos-bucket'` for bucket already exists. Skipping update of ownership and the setting of permissions of the directory for the user with `uid:gid=2002:100`
Bucket `b-watsonx-cos` created successfully

4.5 Define IBM Storage Scale S3 buckets to IBM watsonx.data

Install IBM watsonx.data using the documented procedure at [Installing watsonx.data](#).

Use the following procedure to register a Storage Scale S3 bucket to your watsonx.data instance as externally managed storage and associate a catalog for the bucket. This catalog serves as the query interface for watsonx.data for the data stored within the bucket. Refer to Figure 4-2 on page 31.

The screenshot shows the 'Add component - IBM Storage Scale' configuration window in the IBM watsonx.data console. The window has a dark theme and a sidebar on the left with 'Add component' and 'Configuration' options. The main content area is divided into sections:

- General information:** A 'Display name' field with the example 'Your Storage 01'.
- Storage configuration:**
 - 'Bucket name' field with example 'your-bucket-01'.
 - 'Endpoint' field with format 'http://<hostname or IP>:<port>'.
 - 'Access key' and 'Secret key' fields, both with eye icons for visibility toggles.
 - 'Connection status' section showing 'Untested' with a 'Test connection' button.
- At the bottom, there is a checkbox labeled 'Associate Catalog' with an information icon.

Figure 4-2 watsonX.data panel for adding IBM Storage Scale component

- ▶ Log in to watsonx.data console.
- ▶ From the navigation menu, select Infrastructure Manager.
- ▶ Click Add component.
- ▶ In the Add component window, select the IBM Storage Scale tile and provide the details of the S3 bucket.
 - **Bucket name** - Enter the actual name of the S3 bucket as known to the IBM Storage Scale cluster. This could be a local or an accelerated bucket in IBM Storage Scale.
 - **Display name** - Choose a display name of the bucket.
 - **Endpoint** - Enter the endpoint URL. The URL is in the form of a http(s)://<IP Address>:<port>.
 - For <IP Address>, substitute this with the output of “mmces address list” command as explained in Figure 4-2.
 - Refer to the output of “mms3 config list” command for the port number used by the S3 service, which is 6001 for http and 6443 for https by default.

Note: For higher throughput and performance from the S3 service, a load balancer may be used, which works by distributing the workload among all protocol nodes. If a load balancer is configured, make sure to use the DNS name of the balancer in the endpoint URL instead of using the CES IP directly. For more details on using a load balancer, see IBM Storage Scale documentation [Load balancing](#).

- **Access key** - Enter your access key.
- **Secret key** - Enter your secret key.
- **Connection Status** - Click the Test connection link to test the bucket connection.
- **Associate Catalog** - Select the check box to add a catalog for your storage.
- **Catalog type** - Select the catalog type from the list. The recommended catalog is Apache Iceberg. The other options for catalog are Apache Hive, Apache Hudi, and Delta Lake.
- **Catalog name** - Choose a name for the associated catalog.

To add a bucket-catalog pair, see [Adding a storage-catalog pair](#)

Create an engine instance such as Presto and associate the catalog to that engine. This will make the S3 bucket discoverable through the catalog. Then continue to create schemas and tables under the storage catalog, as shown in the following command:

```
create schema <catalog name>.<schema name> with (location = 's3a://<bucket name>/<directory for schema>');
```

For example, if the name of the bucket is b-watsonx and the catalog name is c_scale, create a schema named “schema1” by running:

```
create schema c_scale.schema1 with (location = 's3a://b-watsonx/schema1');
```

This creates a directory called schema1 under the bucket's filesystem path in

IBM Storage Scale. Data for all tables created under this schema would reside underneath this directory.

4.6 Add a Milvus vector database service to IBM watsonx.data

Add a Milvus vector database service instance from the IBM watsonx.data Infrastructure view GUI panel, see Figure 4-3 on page 33.

- ▶ Select the size of the database based on your infrastructure needs.
- ▶ Choose one of the IBM Storage Scale buckets to be the backing storage for your Milvus vector database.
- ▶ Choose a path under the IBM Storage Scale S3 bucket (such as /milvus01_data) where the vector embeddings will be stored. This way, a common bucket could be used to storage vector embeddings as well as other analytics data such as parquet files. Otherwise, a dedicated Storage Scale bucket could be used for Milvus as well.

The Milvus service, once created, exposes GRPC and REST API services together with a SSL public certificate. Applications from IBM watsonx.ai in turn connect to these endpoints to ingest data and query the vector database.

Add component - Milvus

Select the type of component you want to add to the lakehouse infrastructure and provide the necessary information.

- ⌚ Add component
- ⌚ Configuration

General information

Display name

Example: Your Service 01

Service configuration

Size

Small Recommended for 10 million vectors, 64 index parameters, 1024 segment size and 384 dimensions	Medium Recommended for 50 million vectors, 64 index parameters, 1024 segment size and 384 dimensions
Large Recommended for 100 million vectors, 64 index parameters, 1024 segment size and 384 dimensions	

Add storage bucket ⓘ

Storage bucket

Choose an option

Path

Specify path in the bucket where you want to store vectorized data files. This path will be appended by the autogenerated wxd service id (example: your path/<service-id>).

Figure 4-3 IBM watsonx.data GUI view of the analytics infrastructure

The diagram, Figure 4-4 on page 34, shows a sample view of the watsonx.data Infrastructure manager GUI including multiple engines, services, catalogs and IBM Storage Scale buckets as known to the environment. The diagram showcases the separation of compute, data and metadata layers and highlights the disaggregated nature of the architecture in terms of plug and play enabled modular components.

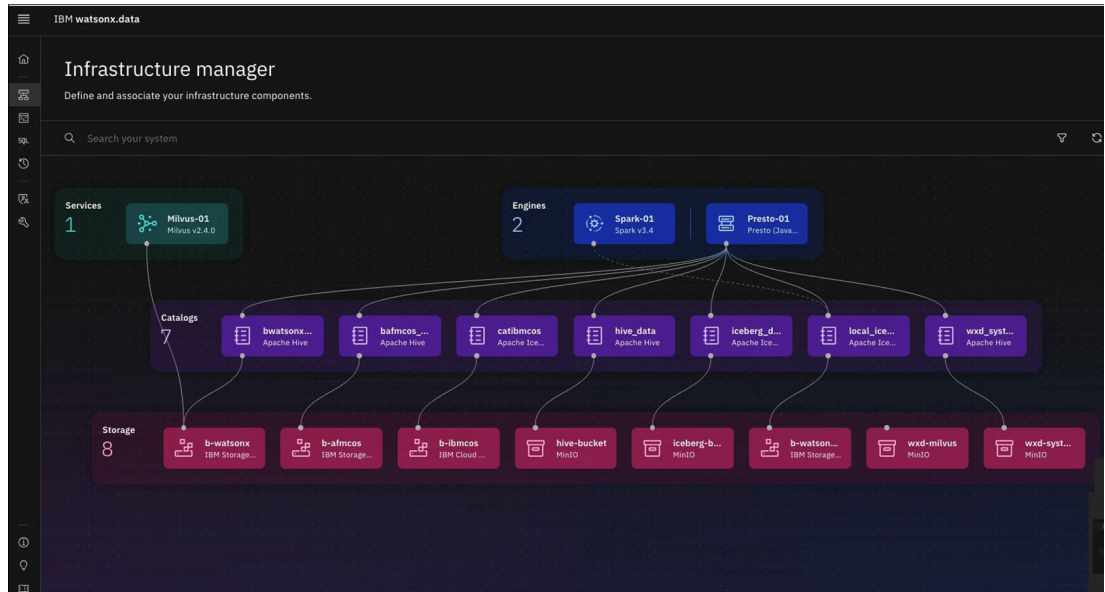


Figure 4-4 Sample view of the watsonx.data Infrastructure manager GUI



Examples, Use Cases and Solution application

AI use cases are vast, and models are developing fast. Large-language models (LLMs) enable use cases that were impossible just a few years ago. One of the biggest problems of LLMs, however, is that they are prone to hallucinations, i.e., making up answers and making up facts. As of the time of this writing, retrieval augmented generation (RAG) is the best technique to address LLM hallucination.

In this chapter, we describe text extraction and RAG use cases and show their implementation for the solution architecture. We employ open tools and benchmarks to demonstrate the approach. We also show an extended concept like role-based content filter and how it can be incorporated into the RAG flow.

In this chapter:

- ▶ 5.1, “Working With IBM watsonx.ai” on page 38
- ▶ 5.2, “Use Case: Robust Entity Extraction and Summarization With Docling” on page 40
- ▶ 5.3, “Use Case: Bulk data ingest and query evaluation with Open RAG Benchmark” on page 45
- ▶ 5.4, “Use Case: RAG with Role-Based Content Filter on IBM Storage Scale” on page 50

5.1 Working With IBM watsonx.ai

In this section, we describe ways to interact and work with IBM watsonx.ai. We demonstrate a simple chat workflow that uses custom data to answer questions. This is the main concept of RAG. For the following, we use the environment setup as described in the previous chapter.

5.1.1 GUI for an Interactive Workflow

IBM watsonx.ai has a built-in chat ability and workbench called Prompt Lab, see Figure 5-1. It is quite powerful and flexible allowing to work with foundation models through basic chat prompts, further interaction options using Prompt Engineering, and even scanning custom data for prompts, for example.

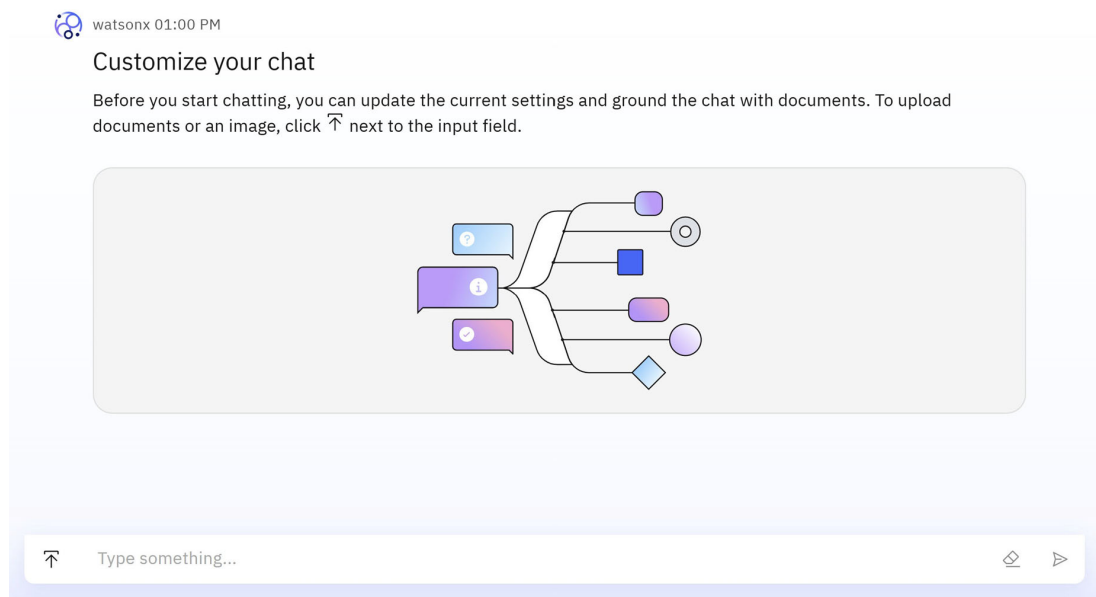


Figure 5-1 Prompt Lab UI

Foundation models can scan and analyze custom data. The Prompt Lab UI allows to upload documents on the fly that are needed to answer a question, see Figure 5-2 on page 39.

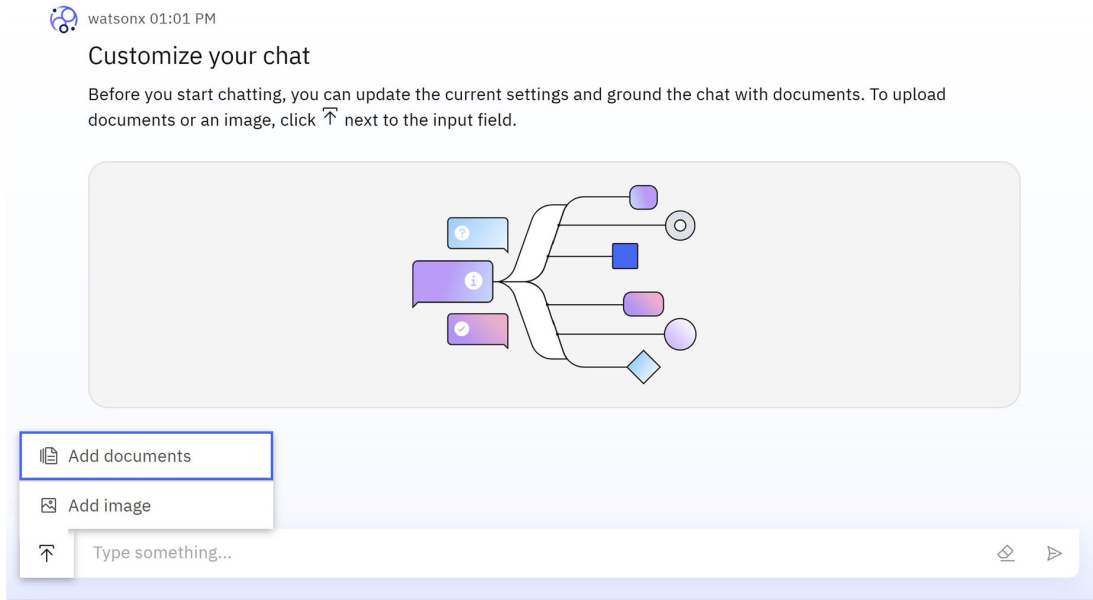


Figure 5-2 Add documents to Prompt Lab

The document being analyzed is stored in memory or can be persistently stored in IBM watsonx.data by default, for example, as illustrated in Figure 5-3.

Ground gen AI with vectorized documents

Add documents to vectorize and create a vector index in memory. Otherwise, select your vector database and specify index details.

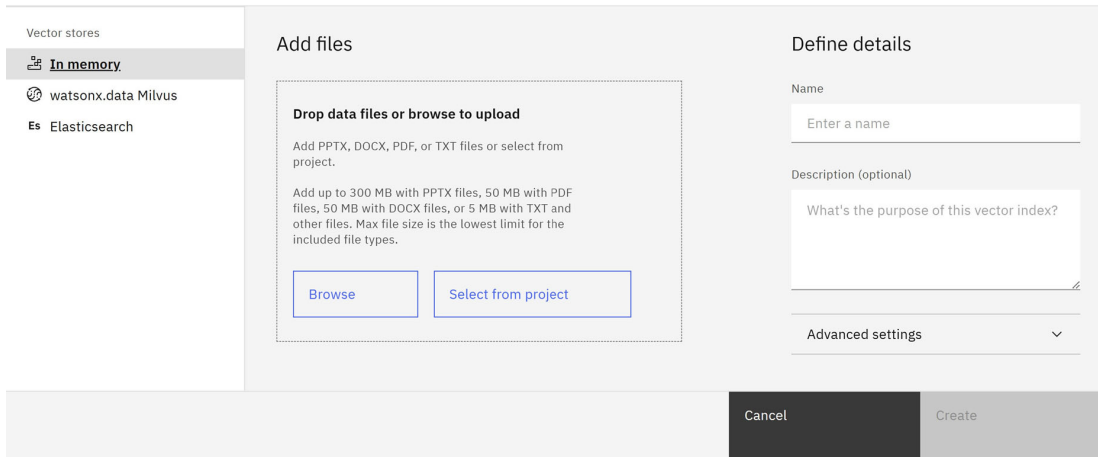


Figure 5-3 Use in-memory, IBM watsonx.data, or other data storage in Prompt Lab

The Prompt Lab UI is helpful for exploration of information or quick testing. It allows quick interactions with foundation models without needing to write a lot of code. It is an easy to understand UI with helpful wizards and the perfect way to take the first steps and even beyond.

5.1.2 Notebook for a programmatic workflow

For repeatable tasks, usually eventually you would want to write code and eventually run a custom application to automate the AI workflow. IBM watsonx.ai has integrated Jupyter notebook support for easy automation and programming of use cases, see Figure 5-4 on

page 40. The notebooks save the code and can be executed in blocks, changed, updated and rerun. The notebooks combine the simplicity of an intuitive UI with automating the AI workflow. This simplifies the automation process without having to directly start writing, deploying, and managing custom applications.

```
[1]: !pip install boto3
!pip install pypdf

Requirement already satisfied: boto3 in /opt/conda/envs/Python-RT24.1-Premium/lib/python3.11/site-packages (1.29.1)
Requirement already satisfied: botocore<1.33.0,>=1.32.1 in /opt/conda/envs/Python-RT24.1-Premium/lib/python3.11/site-packages (from boto3) (1.32.1)
Requirement already satisfied: jmespath<2.0.0,>=0.7.1 in /opt/conda/envs/Python-RT24.1-Premium/lib/python3.11/site-packages (from boto3) (1.0.1)
Requirement already satisfied: s3transfer<0.8.0,>=0.7.0 in /opt/conda/envs/Python-RT24.1-Premium/lib/python3.11/site-packages (from boto3) (0.7.0)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Python-RT24.1-Premium/lib/python3.11/site-packages (from botocore<1.33.0,>=1.32.1->boto3) (2.8.2)
Requirement already satisfied: urllib3<2.1,>=1.25.4 in /opt/conda/envs/Python-RT24.1-Premium/lib/python3.11/site-packages (from botocore<1.33.0,>=1.32.1->boto3) (1.26.19)
Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-RT24.1-Premium/lib/python3.11/site-packages (from python-dateutil<3.0.0,>=2.1->botocore<1.33.0,>=1.32.1->boto3) (1.16.0)
Requirement already satisfied: pypdf in /opt/conda/envs/Python-RT24.1-Premium/lib/python3.11/site-packages (4.2.0)

[2]: import time
start_time = time.time()

Ingest files (1000 PDFs)

[3]: import boto3
import os
```

Figure 5-4 Jupyter Notebook UI example

The code listings in the following sections can be executed directly in such a notebook described in this section.

5.2 Use Case: Robust Entity Extraction and Summarization With Docling

Docling, an open-source project of the Linux Foundation, is a powerful document processing tool that addresses a critical challenge in generative AI applications: extracting and structuring information from diverse document formats. By parsing everything from PDFs and Office documents to audio files and images, Docling provides advanced understanding capabilities including page layout analysis, table structure recognition, and OCR (Optical Character Recognition) support. This is particularly valuable for the context engineering challenge - AI agents/LLMs often need to access information locked in various document formats to build their context.

Docling provides plug-and-play integrations with major AI frameworks like LangChain, LlamaIndex, and Crew AI. Combined with its unified Docling Document format and multiple export options (Markdown, HTML, JSON), it makes an essential bridge between unstructured documents and AI agents. It produces LLM-friendly Markdown while preserving tables, reading order and figures, which improves downstream extraction quality and reduces token cost for LLMs.

Combining a high-throughput S3-compatible storage layer with parallel downloads, with Docling for fast, structure-aware PDF to Markdown conversion, we can create a predictable, secure, and high-performance IBM watsonx.ai inference pipeline. This section describes the architecture, implementation pattern, and a sample implementation for on-premise environments.

5.2.1 High-level Workflow Overview

The high-level workflow realized by the architecture as illustrated in Figure 5-5 and follows these steps:

1. The application receives a processing request.
2. A worker/service lists matching files in IBM Storage Scale using S3 API.
3. Each file is converted by Docling into Markdown - conversion can be CPU/GPU-bound and parallelizable across workers.
4. Markdown is passed to IBM watsonx.ai using a short, deterministic prompt template for either entity extraction or summarization.
5. Results (entities, summary, metadata) are written back to a results store (such as IBM Storage Scale object store) and optionally fed into a vector DB for RAG use-cases.

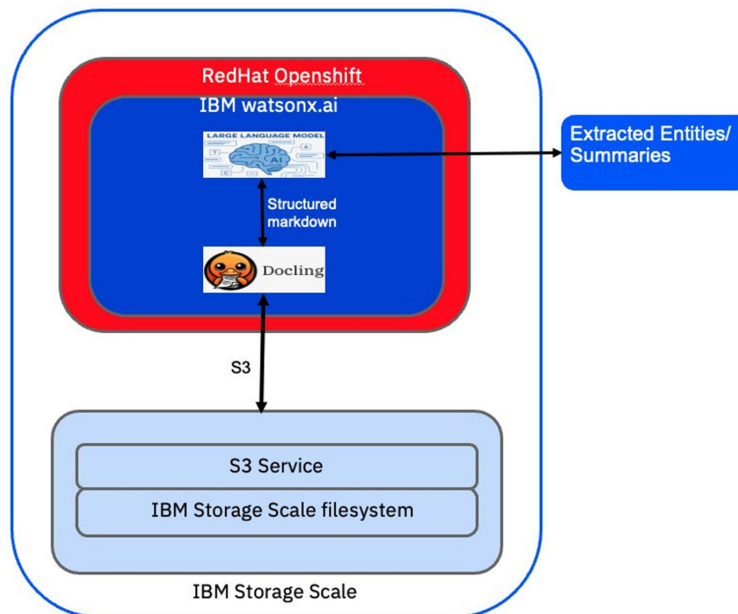


Figure 5-5 Content extraction overview

5.2.2 Financial Summary Example

The following shows a compact Python example that demonstrates the core pieces, i.e. how the data is fetched from storage, and is then processed using Docling and further utilized by an LLM for entity extraction and summarization.

Data preparation

The page shown in Figure 5-6 on page 42 contains tabular data along with other textual content. The table itself does not have visible column borders. Similarly, there can be many variations in how the data is laid out. It is therefore crucial to preserve the original layout before sending it to the LLM, as any loss of structure may cause misinterpretation and lead to inaccurate outputs.



Key Financial Metrics

5

- Gross and pre-tax margin expansion, led by services
- Expense E/R reflects ongoing productivity and lower level of workforce actions, mitigated by lower IP income
- Net income and operating earnings per share reflect significant tax headwind
- Solid free cash flow performance supports investment and shareholder returns

P&L Highlights	1Q19	B/(W) Yr/Yr	P&L Ratios - Operating	1Q19	B/(W) Yr/Yr
Revenue	\$18.2	(1%)	Gross Profit Margin	44.7%	0.9 pts
Cloud & Cognitive Software	\$5.0	2%	Expense E/R	32.4%	2.2 pts
Global Business Services	\$4.1	4%	Pre-Tax Income Margin	12.3%	3.2 pts
Global Technology Services	\$6.9	(3%)	Tax Rate	10.0%	(40.6 pts)
Systems	\$1.3	(9%)	Net Income Margin	11.0%	(0.9 pts)
Pre-Tax Income - Operating	\$2.2	28%	Cash Highlights		LTM
Net Income - Operating	\$2.0	(12%)	Free Cash Flow (excl. GF Receivables)	\$1.7	\$12.2
Earnings Per Share - Operating	\$2.25	(8%)	Share Repurchase (Gross)	\$0.9	\$4.6
			Dividends	\$1.4	\$5.7
			Cash Balance @ March 31	\$18.1	

Revenue growth rates @CC, \$ in billions

Figure 5-6 Financial summary document example

The following code listing demonstrates how, with just a few lines of code, the above unstructured documents can be converted into Markdown format while preserving their original layout.

Example 5-1 Sample code to convert a pdf document into Markdown

```

from docling.document_converter import DocumentConverter

source = " IBM-Earnings-data.pdf "

converter = DocumentConverter()
result = converter.convert(source)

markdown_content= result.document.export_to_markdown()

# Print Markdown
print(markdown_content)

```

The Docling output in Markdown format is shown in Figure 5-7 on page 43.

Key Financial Metrics

- □ Gross and pre-tax margin expansion, led by services
- □ Expense E/R reflects ongoing productivity and lower level of workforce actions, mitigated by lower IP income
- □ Net income and operating earnings per share reflect significant tax headwind
- □ Solid free cash flow performance supports investment and shareholder returns

P&L Highlights	1Q19	B/(W) Yr/Yr
Revenue	\$18.2	(1%)
Cloud& Cognitive Software	\$5.0	2%
Global Business Services	\$4.1	4%
Global Technology Services	\$6.9	(3%)
Systems	\$1.3	(9%)
Pre-Tax Income - Operating	\$2.2	28%
Net Income - Operating	\$2.0	(12%)
Earnings Per Share - Operating	\$2.25	(8%)

P&L Ratios - Operating	1Q19	B/(W) Yr/Yr
Gross Profit Margin	44.7%	0.9 pts
Expense E/R	32.4%	2.2 pts
Pre-Tax Income Margin	12.3%	3.2 pts
Tax Rate	10.0%	(40.6 pts)
Net Income Margin	11.0%	(0.9 pts)
Cash Highlights		LTM
Free Cash Flow (excl. GF Receivables)	\$1.7	\$12.2
Share Repurchase (Gross)	\$0.9	\$4.6
Dividends	\$1.4	\$5.7
Cash Balance @March31	\$18.1	

Figure 5-7 Financial summary transformed to Markdown

LLM setup and prompt for extracting entities from the document

The processed document can be utilized by an LLM to extract structured entities using a well-crafted prompt as follows.

Example 5-2 code to initialize the LLM

```
from ibm_watsonx_ai.metanames import GenTextParamsMetaNames as GenParams
from ibm_watsonx_ai.foundation_models.utils.enums import DecodingMethods
from langchain_ibm import WatsonxLLM

parameters = {
    GenParams.DECODING_METHOD: DecodingMethods.SAMPLE.value,
    GenParams.MAX_NEW_TOKENS: 2000,
    GenParams.MIN_NEW_TOKENS: 1,
    GenParams.TEMPERATURE: 0.5,
    GenParams.TOP_K: 50,
    GenParams.TOP_P: 1
}

llm = WatsonxLLM(
    model_id="ibm/granite-3-2-8b-instruct",
    url=credentials["url"],
    apikey=credentials["apikey"],
    project_id=project_id,
    params=parameters
)
```

Usage Example:

LLM prompt for extracting entities from the pdf leveraging the docling output.

Example 5-3 LLM instruction example

```
template="""You are a skilled Financial Assistant responsible for analyzing
equity research reports and extract important entities.
```

```
You will be given an equity research report in markdown format. Extract the
following entities for quarter 1 of 2019 and present them in json format:
```

```
Revenue for Q1, 2019 in $ billions
- Gross profit margin for Q1, 2019
```

```
-----
OUTPUT FORMAT (STRICT)
-----
```

```
Return ONLY valid JSON text in the exact structure below:
```

```
---
{
  "Revenue" : "x.x",
  "Gross profit margin" : "x.x%"
}
---
```

Rules:

- The output must be valid JSON with no text outside JSON.
- All numbers must be strings (e.g., "7.5").

Markdown report:

```
{{markdown_data}}
```

Json output: ""

```
prompt = PromptTemplate(
  template=template,
  input_variables=['markdown_data'],
  template_format="jinja2"
)
prompt_text = prompt.format(markdown_data=markdown_content)
model_output_text = None
gen = llm.generate([prompt_text], temperature=0)
print(gen.generations[0][0].text)
```

Output from IBM watsonx.ai LLM:

Finally, we have the require information extracted by the LLM and provided in the format requested.

Example 5-4 watsonx output returned

```
{
  "Revenue": "18.2",
  "Gross profit margin": "44.7%"
}
```

Further extensions

For more complex scenarios, the above example can be extended. For example, for pdfs containing images, we can use the multi-modal VLM (visual language model) pipeline from docling.

Example 5-5 sample code to use a Granite® VLM model to convert content

```

from docling.datamodel.base_models import InputFormat
from docling.document_converter import DocumentConverter, PdfFormatOption
from docling.pipeline.vlm_pipeline import VlmPipeline
from docling.datamodel.pipeline_options import (
    VlmPipelineOptions,
)
from docling.datamodel import vlm_model_specs

pipeline_options = VlmPipelineOptions(
    vlm_options=vlm_model_specs.GRANITEDOCLING_TRANSFORMERS,
)

converter = DocumentConverter(
    format_options={
        InputFormat.PDF: PdfFormatOption(
            pipeline_cls=VlmPipeline,
            pipeline_options=pipeline_options,
        ),
    }
)

doc = converter.convert(source=pdf_path).document
markdown_content= doc.export_to_markdown()

```

The above code listings demonstrate how combining IBM Storage Scale with a robust Docling-powered data pipeline enables IBM watsonx.ai LLM to produce more accurate results. This pattern keeps each step focused and horizontally scalable: IBM Storage Scale handles storage throughput; Docling standardizes documents into LLM-friendly Markdown; and IBM watsonx.ai performs extraction and summarization. Together, they enable a predictable performance for bulk file processing use cases.

5.3 Use Case: Bulk data ingest and query evaluation with Open RAG Benchmark

This section demonstrates a RAG use case to provide LLMs with new information. To this end, we use the Vectara Open RAG Benchmark available at <https://github.com/vectara/open-rag-bench>. The benchmark contains a dataset of 1000 scholarly articles in PDF format. In addition, it comprises more than 3000 queries to use for RAG testing. Out of the 1000 documents, 400 contain the answers of the queries, while the remaining 600 documents are noise and irrelevant to the queries.

For this paper, we employ the Open RAG Benchmark as an API rather than a benchmark to model the functional process and to show a bulk data ingest workflow. We will load the 1000 documents into the RAG pipeline and store them retrievable into its knowledge base, i.e., the vector database.

5.3.1 Data Ingest

The data ingest process first pulls all the files. The files are parsed and partitioned into chunks. For each chunk, corresponding embeddings, which are the suitable vector

representation, are created. The chunks, together with their embeddings, are stored in the vector database.

Pull files

We use an S3 client to list all files from an S3 bucket on IBM Storage Scale:

Example 5-6

```
s3_client = boto3.client('s3', endpoint_url='[URL]',
                        aws_access_key_id='xxxxxx',
                        aws_secret_access_key='yyyyyyy')

bucket="b-scale-wxai-data"
folder="open_ragbench/PDFs/"
response = s3_client.list_objects_v2(Bucket=bucket, Prefix=folder, Delimiter="/")
```

Iterating through the list, we download all PDFs and store the filenames:

```
files = []
for fileKey in response['Contents']:
    file = fileKey['Key']
    if (file.endswith(".pdf")):
        target = os.path.basename(file)
        s3_client.download_file(bucket, file, target)
        files.append(target)
```

Parsing and chunking

We use the downloaded files and use a standard library to parse the files and use a text splitter for chunking:

Example 5-7

```
passages = []
for file_path in files:
    # process the file
    loader = PyPDFLoader(file_path)
    text_splitter = RecursiveCharacterTextSplitter(
        chunk_size=512,
        chunk_overlap=20,
        length_function=len,
        is_separator_regex=False,
    )
    pages = loader.load_and_split(text_splitter)
    for page in pages:
        passages.append(page.page_content)
```

Create embeddings

Next we iterate through the chunks. We use an embedding model provided by IBM watsonx.ai to create vector embeddings. Interestingly enough, the embedding model runs on the CPU and does not require a dedicated GPU.

Example 5-8

```
from ibm_watsonx_ai.foundation_models.embeddings import
SentenceTransformerEmbeddings
model = SentenceTransformerEmbeddings("all-MiniLM-L6-v2")
```

```

embeddings = []
for passage in passages:
    embedding = model.embed_query(passage)
    embeddings.append(embedding)

```

Ingest into Vector DB

Finally to ingest the data into the vector database, we first establish a database connection. We use the Milvus client library for that purpose.

Example 5-9

```

uri = "https://[host]:[port]" # Construct URI from host and port
user = 'xxxxxx'
password = 'yyyyyy'
client = MilvusClient(uri=uri, user=user, password=password,
                      secure=True,
                      db_name="default",
                      server_pem_path="milvus.pem",
                      # digital certificate file
                      server_name="[host]")

```

Once the connection is created, we define the data schema for the collection that stores the data:

Example 5-10

```

collection_name = "vectara_bench"
primary_key = FieldSchema(
    name="id",
    dtype=DataType.INT64,
    is_primary=True,
)

vector = FieldSchema(
    name="vector",
    dtype=DataType.FLOAT_VECTOR,
    dim=384,
)

passage_text = FieldSchema(
    name="passage",
    dtype=DataType.VARCHAR,
    max_length=2048
)

schema = CollectionSchema(
    fields = [primary_key, vector, passage_text]
)

```

When the schema is defined, we create the data collection and then iterate through the chunks to insert them in the database:

Example 5-11

```

client.create_collection(
    collection_name=collection_name,

```

```

        schema=schema,
        using="default"
    )
    id = 0
    for (passage, embed) in zip( passages, embeddings ):
        data=[{"id": id, "vector": embed, "passage": passage}]
        res = client.insert(
            collection_name=collection_name,
            data=data
        )
        id += 1

```

5.3.2 RAG Query

To demonstrate the RAG-enhanced query, we next show setting up the model and loading the data collection. Afterwards, we take one of the Open RAG Benchmark queries as example and show the RAG-enhanced answer.

Model and collection setup

To work with the models provided by IBM watsonx.ai, we need to setup an authentication mechanism. We then use the Granite model, which runs on a GPU. The authentication and model instantiation is shown here:

Example 5-12

```

wslib = access_project_or_space()
def get_credentials():
    return {
        "url" : os.getenv('RUNTIME_ENV_APSX_URL'),
        "token" : wslib.auth.get_current_token(),
        "instance_id": "openshift"
    }

model_id = "ibm/granite-3-8b-instruct"
parameters = {
    "decoding_method": "greedy",
    "max_new_tokens": 350,
    "repetition_penalty": 1
}
project_id = os.getenv("PROJECT_ID")
space_id = os.getenv("SPACE_ID")

llm = Model(
    model_id = model_id,
    params = parameters,
    credentials = get_credentials(),
    project_id = project_id,
    space_id = space_id
)

```

To operate on the data of the vector database, we load the respective collection as follows:

Example 5-13

```

index_params = MilvusClient.prepare_index_params()

```

```

index_params.add_index(
    field_name="vector",
    metric_type="L2",
    index_type="IVF_FLAT",
    index_name="vector_index",
    params={ "nlist": 128 }
)

client.create_index(
    collection_name=collection_name,
    index_params=index_params
)

client.load_collection(collection_name=collection_name)

```

Query example

The Open RAG Benchmark contains many queries. For this paper, we show one of these queries:

Example 5-14

```

with open(queries_file) as f:
    queries_json = json.load(f)

# example query
question = queries_json["9199173b-3ed1-4118-88cd-1713fc5fa8a7"]
print(question)

```

Output:

```
{'query': 'How do changes in effective microbial death rate influence parameters like alpha and beta?', 'type': 'abstractive', 'source': 'text'}
```

To obtain the RAG-enhanced answer, we first create the embedding of the question and then use this to retrieve the relevant context. The question together with the context builds the RAG prompt that is sent to the LLM to obtain the result:

Example 5-15

```

question_embedding = model.embed_query(question)

num_results = 5
search_params = {
    "metric_type": "L2",
    "params": {"nprobe": 5}
}
res = client.search(
    collection_name=collection_name,
    data=[question_embedding],
    anns_field="vector",
    search_params=search_params,
    limit=num_results,
    output_fields=["passage", "role"],
)

```

```

relevant_chunks = []
for i in range(num_results):
    text = res[0][i].get('entity').get('passage')
    relevant_chunks.append(text)
context = "\n\n".join(relevant_chunks)

rag_prompt = f"Please answer a question using the following context. " + \
f"If the question is unanswerable, say \"unanswerable\"." + \
f"If the answer can be found in a paragraph, include the paragraph in the response." + \
f"\n\nContext:\n\n{context}\n\n" + \
f"\n\nQuestion: {question}"
rag_response = llm.generate_text(prompt=rag_prompt, guardrails=False)
print(f"Question: {question}")
print(rag_response)

```

Output:

Example 5-16 RAG enhanced answer

Question: How do changes in effective microbial death rate influence parameters like alpha and beta?

Answer: Changes in effective microbial death rate, represented by the decay rate ratio (β), have a significant influence on parameters like alpha. As the noise level increases, both alpha and beta decrease, indicating system instability. However, for stable oscillation, a middle level of decay rate ratio (β) is required. This is demonstrated by both indicators increasing to a peak at $\beta \cdot 1.8 \pm 0.2$ and then decreasing with higher β . This pattern is consistent across various noise levels, as shown in Figure S5.

Reference(s):

S3.1 Numerical procedure

As the parametrization $x(?)$ is not available analytically, for each set of rate noise levels and decay rate ratios (Figs.2C, D). Both indicators consistently demonstrate a clear trend with changing parameters. Firstly, as the noise level increases, both indicators decrease, suggesting that the system becomes unstable. Secondly, both indicators exhibit a pattern of increasing to a peak at $\beta \cdot 1.8 \pm 0.2$ and then decreasing with higher β . This demonstrates that stable oscillation requires a middle level of decay rate ratio. Similar results under distinct noise levels are shown in Fig. S5.

The above output shows the question and the RAG-enhanced answer. It also contains references to support the result. The example here is highly domain specific, as are typical RAG use cases.

5.4 Use Case: RAG with Role-Based Content Filter on IBM Storage Scale

This section extends the previous use case by a multi-tenant approach. Typically, the core knowledge base in a RAG pipeline contains a lot of information, not all of which is of interest for all users. One way to address this issue is to shard the database and split it along the information, however, this can lead to lots of databases, potential data duplicates, and

manageability issues. Another approach, demonstrated in the following, enhances the database and the RAG pipeline with a role-based content filter. We highlight the specific extensions to the implementation of the previous use case.

5.4.1 Data Formats and Data Preparation

One key distinction in this approach is the data format itself. It is annotated with the roles allowed to access the data. Next, we will show how the data is loaded into the system, specific data encoding for the roles, and the vector database schema.

Data input

First the data is pulled normally with an S3 client:

Example 5-17 S3 data call

```
s3_client.download_file(bucket, file, target)
```

Next, the data is annotated specifically with the corresponding roles allowed to access the data, where access by multiple roles is also allowed. Hence, we use an array to encode the list of roles and, for this example, use 'T' for a Technical role, 'H': HR role. The encoding can be done per ingest stream or with individual logic. Here, we use two PDF files, one with salary information of the company and second an IBM Storage Scale System data sheet containing technical specifications. We annotate the files directly, then we parse and chunk them:

Example 5-18 File parsing code

```
files = {
    "Information-Technology,roles-and-salaries.pdf": ["H"],
    "ibm-ess-data-sheet.pdf": ["T"],
}
passages = []
for file_path in files:
    role = files[file_path]
    # process the file
    loader = PyPDFLoader(file_path)
    text_splitter = RecursiveCharacterTextSplitter(
        chunk_size=512,
        chunk_overlap=20,
        length_function=len,
        is_separator_regex=False,
    )
    pages = loader.load_and_split(text_splitter)
    for page in pages:
        passages.append([page.page_content, role])
```

Data encoding

When we insert the data and their vector embeddings in the vector database, we need to maintain the roles information. We encode this information with a bitmask, 01b is used for access by the technical role, 10b is used for access by the HR role, and 11b is used for access by both roles. This way the approach can be easily extended for more roles:

Example 5-19 Embedding code

```
model = SentenceTransformerEmbeddings("all-MiniLM-L6-v2")
embeddings = []
```

```

for passage in passages:
    embedding = model.embed_query(passage[0])
    embedding_value=0
    if 'T' in passage[1]:
        embedding_value+=1
    if 'H' in passage[1]:
        embedding_value+=2
    embeddings.append([embedding, embedding_value])

```

Vector database collection definition

For the schema of the vector database collection, we extend the regular collection with a “role” field when we create the collection:

Example 5-20 adding role field and creating collection

```

collection_name = "rb_rag"
primary_key = FieldSchema(
    name="id",
    dtype=DataType.INT64,
    is_primary=True,
)

vector = FieldSchema(
    name="vector",
    dtype=DataType.FLOAT_VECTOR,
    dim=384,
)

passage_text = FieldSchema(
    name="passage",
    dtype=DataType.VARCHAR,
    max_length=2048
)

role = FieldSchema(
    name="role",
    dtype=DataType.INT64
)

schema = CollectionSchema(
    fields = [primary_key, vector, role, passage_text]
)

client.create_collection(
    collection_name=collection_name,
    schema=schema,
    using="default"
)

```

5.4.2 Model Setup

The model collection remains the same as the regular RAG pipeline:

Example 5-21 Model setup code

```

model_id = "ibm/granite-3-8b-instruct"
parameters = {
    "decoding_method": "greedy",
    "max_new_tokens": 350,
    "repetition_penalty": 1
}

llm = Model(
    model_id = model_id,
    params = parameters,
    credentials = get_credentials(),
    project_id = project_id,
    space_id = space_id
)

```

5.4.3 RAG Queries

For the demonstration of the RAG answers, we show two examples in the following.

Technical role

First we use an example for the technical role. We want to know the important features of the IBM Storage Scale 6000. We pass the value of the technical role and use this when we search for the relevant context in the vector database:

Example 5-22 User role addition

```

user_role = 1 # T
question = "What are the features of ESS 6000?"
question_embedding = model.embed_query(question)

num_results = 5
search_params = {
    "metric_type": "L2",
    "params": {"nprobe": 5}
}
res = client.search(
    collection_name=collection_name,
    data=[question_embedding],
    anns_field="vector",
    search_params=search_params,
    filter=f"role % 2 == {user_role}",
    limit=num_results,
    output_fields=["passage","role"],
)

```

The search filter controls that we can only retrieve the data that is relevant for the technical role. Based on the context of the RAG query, we build the RAG prompt. We show the RAG prompt itself to demonstrate this:

Example 5-23 RAG prompt creation

```

relevant_chunks = []
for i in range(num_results):

```


supports the NVIDIA GPUdirect Storage protocol, which enables a direct data path between GPU memory and local or remote storage, such as NVMe or NVMe over Fabric (NVMe-oF). This GPUdirect"

Question: What are the features of ESS 6000?

With this prompt, we can query the LLM and obtain our result:

Example 5-25 LLM query call

```
rag_response = llm.generate_text(prompt=rag_prompt, guardrails=False)
print(f"Question: {question}")
print(rag_response)
```

Output:

Example 5-26

Question: What are the features of ESS 6000?

Answer: The IBM Storage Scale System 6000, also known as ESS 6000, offers several features. It can be configured with standard NVMe flash drives for maximum performance or with IBM FlashCore Modules for higher data density and compression. The system provides up to 310 gigabytes per second (GB/S) throughput with low latency and up to 13 millions IOPS using NVMeoF. It has up to 1.8PBe (effective capacity) in a standard 4U rack space. The Storage Scale software enables the system to scale linearly, with throughput increasing proportionally as more systems are added to a cluster. For sequential data and workloads requiring access to massive data sets, ESS 6000 supports up to nine expansion enclosures. The IBM Storage Scale Expansion Enclosure contains up to 91 20TB or 22TB self-encrypting SAS hard disk drives (HDDs). Attaching eight expansion enclosures to the Storage Scale System 6000 expands the maximum storage capacity to 18PB of HDD. ESS 6000 is a certified storage solution for NVIDIA DGX SuperPOD and supports the NVIDIA GPUdirect Storage protocol, enabling a direct data path between GPU memory and local or remote storage, such as NVMe or NVMe over Fabric (NVMe-oF).

Reference(s):

Solution Brief – IBM Storage Scale & Storage Scale System

Figure 1 – The IBM Storage Scale System 6000 can deliver up to 310GB/s throughput,

As we had hoped, we can answer the question with the relevant context.

HR role

In the next example, we want to address a similar problem from the HR role's perspective. Let's assume this may entail access to sensitive information, like company salary data. We want to know what the role of an IT System Administrator is:

Example 5-27 User role addition

```
user_role = 2 # H
question = "What is the role of an IT System Administrator?"
question_embedding = model.embed_query(question)
```

```

num_results = 3
search_params = {
    "metric_type": "L2",
    "params": {"nprobe": 5}
}
res = client.search(
    collection_name=collection_name,
    data=[question_embedding],
    anns_field="vector",
    search_params=search_params,
    filter=f"role >= {user_role}",
    limit=num_results,
    output_fields=["passage","role"],
)

```

When we search for relevant information, again the filter ensures we get the data according to our role. We use this to create the RAG prompt shown in the following:

Example 5-28 RAG prompt creation

```

relevant_chunks = []
for i in range(num_results):
    text = res[0][i].get('entity').get('passage')
    text = text.replace("\n\n", "\n")
    relevant_chunks.append(text)
context = "\n\n".join(relevant_chunks)

rag_prompt = f"Please answer a question using only the following context. " + \
f"If the question is unanswerable, say \"unanswerable\"." + \
f"If the answer can be found in a paragraph, include the paragraph in the\n\nContext:\n\n{context}\n\n" + \
f"\n\nQuestion: {question}"

print(rag_prompt)

```

Output:

Example 5-29 RAG prompt output and context

Please answer a question using only the following context. If the question is unanswerable, say "unanswerable". If the answer can be found in a paragraph, include the paragraph in the response.

Context:

IT System Administrator

- Responsibility for managing information systems in the company.
- Analysing company needs regarding its information systems.
- Designing and implementing information systems.
- Providing technical support to users.
- Collecting and recording requirements for change.

2,869 - 6,326 EUR

IT Tester

- Responsibility for the testing of software applications with an emphasis on detecting errors in terms of functionality, usability, and safety of software.

ICT-specialist

- Providing technical support and troubleshooting for hardware and software issues.
- Implementing and maintaining IT infrastructure including networks, servers, and cloud services.
- Assisting in the development and deployment of new IT solutions and systems.
- Monitoring and optimizing system performance and security.
- Collaborating with cross -functional teams to integrate technology into business processes.
- Conducting regular system audits and assessments to ensure compliance with industry standards.
- Training staff on new technologies and software applications.
- Developing and maintaining documentation for IT processes and procedures.
- Managing user accounts, permissions, and access controls.
- Staying updated with the latest technology trends and advancements to recommend improvements.
- Ensuring data backup and disaster recovery processes are in place and effective."

Question: What is the role of an IT System Administrator?

Finally, we use the RAG prompt to get the answer to the question from the pipeline:

Example 5-30 RAG response code

```
rag_response = llm.generate_text(prompt=rag_prompt, guardrails=False)
print(f"Question: {question}")
print(rag_response)
```

Output:

Example 5-31

Question: What is the role of an IT System Administrator?

Answer: The role of an IT System Administrator involves managing information systems in the company, analyzing company needs regarding its information systems, designing and implementing information systems, providing technical support to users, and collecting and recording requirements for change. The salary range for this role is 2,869 - 6,326 EUR

We see here that we get an answer to the question as well as a salary range information. This can be used, for example, for aiding job interviews in the company.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this paper.

IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- ▶ *IBM Storage Scale System Introduction Guide*,
<https://www.redbooks.ibm.com/abstracts/redp5729.htm>
- ▶ Accelerating AI and Analytics with IBM watsonx.data and IBM Storage Scale,
<https://www.redbooks.ibm.com/abstracts/redp5743.htm>

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

[ibm.com/redbooks](https://www.ibm.com/redbooks)

Online resources

These websites are also relevant as further information sources:

- ▶ IBM watsonx.ai

<https://www.ibm.com/products/watsonx-ai>

- ▶ IBM watsonx.ai product documentation

<https://www.ibm.com/docs/en/watsonx/w-and-w/2.2.0>

- ▶ IBM watsonx.data

<https://www.ibm.com/products/watsonx-data>

- ▶ IBM Documentation for IBM watsonx.data

<https://cloud.ibm.com/docs/watsonxdata>

- ▶ IBM Storage Scale

<https://www.ibm.com/products/storage-scale>

- ▶ Product Documentation for IBM Storage Scale

<https://www.ibm.com/docs/en/storage-scale>

- ▶ Product Documentation for IBM Storage Scale System

<https://www.ibm.com/docs/en/storage-scale-system>

- ▶ IBM Fusion Content-Aware Storage

<https://www.ibm.com/docs/en/fusion-software/2.11.0?topic=services-content-aware-storage-cas>

► Docling

<https://www.docling.ai/>

<https://github.com/docling-project/docling>

Help from IBM

IBM Support and downloads

ibm.com/support

Services from IBM Consulting

ibm.com/services

IBM Training

ibm.com/training

Index

R

Redbooks website 59
Contact us ix



REDP-5765-00

ISBN

Printed in U.S.A.

Get connected

