O'REILLY®

2nd Edition

The Enterprise Data Catalog

Scale AI with Metadata Using LLMs, MCP, and Agentic Architecture

Early Release

RAW & UNEDITED

Compliments of

ACTIAN™

a division of HCLSoftware

Ole Olesen-Bagneux

ACTIAN™
a division of HCLSoftware

The Enterprise Data Catalog

Is Becoming the Context Layer for AI

Modern AI systems need more than access to data. They need trusted metadata, governance, lineage, and business context.

Actian helps organizations build governed, explainable, AI-ready data foundations across hybrid, cloud, and on-premises environments.

- Federated Knowledge Graph
- Unified Catalog and Marketplace
- AI-powered Stewardship and Automation
- End-to-End Lineage
- Semantic Governance and Business Glossaries
- Data Contracts and Observability
- Conversational Analytics and AI Access
- MCP and Agentic AI Integrations

Build the semantic foundation for enterprise AI.

Explore Actian Data Intelligence at actian.com

SECOND EDITION

The Enterprise Data Catalog

Scale AI with Metadata Using LLMs, MCP, and Agentic Architecture

With Early Release ebooks, you get books in their earliest form—the author's raw and unedited content as they write—so you can take advantage of these technologies long before the official release of these titles.

Ole Olesen-Bagneux

O'REILLY®

The Enterprise Data Catalog
by Ole Olesen-Bagneux

Copyright © 2027 O'Reilly Media, Inc. All rights reserved.

Published by O'Reilly Media, Inc., 141 Stony Circle, Suite 195, Santa Rosa, CA 95401.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*https://oreilly.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Acquisitions Editor: Aaron Black
Development Editor: Sara Hunter
Production Editor: Katherine Tozer

Interior Designer: David Futato
Interior Illustrator: Kate Dullea

February 2023: First Edition
June 2027: Second Edition

Revision History for the Early Release
2026-02-18: First Release
2026-03-20: Second Release
2026-06-05: Third Release

See *https://oreilly.com/catalog/errata.csp?isbn=9798341672949* for release details.

The O'Reilly logo is a registered trademark of O'Reilly Media, Inc. *The Enterprise Data Catalog*, the cover image, and related trade dress are trademarks of O'Reilly Media, Inc.

The views expressed in this work are those of the author and do not represent the publisher's views. While the publisher and the author have used good faith efforts to ensure that the information and instructions contained in this work are accurate, the publisher and the author disclaim all responsibility for errors or omissions, including without limitation responsibility for damages resulting from the use of or reliance on this work. Use of the information and instructions contained in this work is at your own risk. If any code samples or other technology this work contains or describes is subject to open source licenses or the intellectual property rights of others, it is your responsibility to ensure that your use thereof complies with such licenses and/or rights.

This work is part of a collaboration between O'Reilly and Actian. See our statement of editorial independence.

979-8-341-67290-1

[LSI]

Table of Contents

Brief Table of Contents (*Not Yet Final*) vii

Preface ix

1. Introduction to Data Catalogs 17
The AI Data Catalog and as Source for AI 18
The Core Functionality of a Data Catalog 19
Create an Overview of the Data in the IT Landscape 20
You Will Always Be Able to See More Metadata than Data 21
Organize Data 21
Enable Search of Company Data 25
Data Discovery 29
The Data Discovery Team 31
Data Catalog Ownership 32
End-User Roles and Responsibilities 33
Summary 35

2. Organize Data: Design a Robust Architecture for Search 37
Using AI to Organize Data 38
Organizing Domains in the Data Catalog 39
Domain Architecture in a Data Catalog 39
Understanding Domains 42
Processes and Capabilities 45
Data Sources 49
Organizing Data in the Domains 51
Metadata for Data 51
Knowledge Graph Powered Data Catalogs 56
Data Assets and Data Products 57

Metadata Quality 60
Classification 64
Summary 67

3. Search For Data: Concepts, Features, Mechanics. 69
Concepts 70
Searching in Data Versus Searching for Data 70
Information Needs: Search Like Librarians—Not Like Data Scientists 75
Serendipity 77
Promptism 78
Features: Search Features in a Data Catalog 79
Simple Search 81
Browsing 84
Complex Search 87
Conversational Search 89
Mechanics: The Mathematics Behind Search 90
Recall and Precision 91
Zipf's Law 94
Summary 96

4. Search For Data Patterns. 99
Search Pattern Overview 99
Keyword Search and Conversational AI Search 100
Basic Simple Search 101
Detailed Simple Search 102
Flexible Simple Search 104
Range Search 105
Block Search 106
Statement Search 110
Browsing Patterns 111
Glossary Browsing 111
Domain Browsing 112
Lineage Browsing 113
Graph Browsing 113
Searching a Graph-Based Data Catalog 115
Summary 116

Brief Table of Contents (*Not Yet Final*)

Preface (available)

Chapter 1: Introduction to Data Catalogs (available)

Chapter 2: Organize Data: Design a Robust Architecture for Search (available)

Chapter 3: Search Data: Concepts, Features, Mechanics, Patterns (available)

Chapter 4: Search for Data Patterns (available)

Chapter 5 Access and Observe Data (unavailable)

Chapter 6 Empower End Users and Engage Stakeholders (unavailable)

Chapter 7 Data Domains (unavailable)

Chapter 8 Data Architecture, Providers, and Consumers (unavailable)

Chapter 9 Data Products and Data Contracts (unavailable)

Chapter 10 Manage Data: Improve Lifecycle Management (unavailable)

Chapter 11 The Data Catalog Is Now a Source in Itself (unavailable)

Chapter 12 The LLM + KG Pattern (unavailable)

Chapter 13 Standards and AI (unavailable)

Preface

A Note for Early Release Readers

With Early Release ebooks, you get books in their earliest form—the author's raw and unedited content as they write—so you can take advantage of these technologies long before the official release of these titles.

This will be the Preface of the final book.

If you'd like to be actively involved in reviewing and commenting on this draft, please reach out to the editor at *shunter@oreilly.com*.

It's A New Day in Search

"So what's your take on ChatGPT? How does it change data catalogs? Do you have an opinion about that?"

Matt Housley was putting me on the spot. I was invited on the podcast Monday Morning Data Chat[1] and we were discussing my soon to be published book *The Enterprise Data Catalog*, the first edition of this book that you are now reading in the second edition.

I wasn't properly mic'ed up, I was not yet used to being on tech podcasts and painfully aware of my clunky sound. I pulled myself together and managed to answer. I said that my book ended with a future vision for data catalogs, and that if I was to go into more depth about that future vision then, obviously I would look into the potential of ChatGPT and AI in general.

1 Monday Morning Data Chat was a podcast by Joe Reis and Matt Housley, I was interviewed in the episode The Future of Data Catalogs, Matt asks the question 52:10

It was a paradoxical moment, really. As I was publishing the first edition of *The Enterprise Data Catalog*, in early 2023, the core message of the book was that data catalogs are like search engines, just for data in companies. That's why data catalogs need to be built on knowledge graphs, just like search engines are.

Don't worry, we will get to knowledge graphs, they play a major role in this book!

And yet, search engines themselves, the real ones, for the web, were unexpectedly challenged, right there in early 2023, as I published *The Enterprise Data Catalog*.

For 25 years, the biggest business on the web - search - had been completely stable. Not technologically, of course it had evolved, but in terms of how you searched the web, as an end user. You had one, simple search bar, that would provide the best, the freshest, the most relevant search hits to you, with the blink of an eye. For 25 years, we had been using search engines as the most natural extension of our mind to search for everything from our absolute basic needs to the most complex types of curiosity the human brain can produce. And then, all of a sudden, it was over.

In November 2022, ChatGPT 3.5 was released by OpenAI.

Suddenly, we asked ourselves if the way we search was in fact obsolete. Would we talk with the web instead? Do *conversational search* through a chatbot?

In those very early months of 2023 Microsoft CEO Satya Nadella said: *"It's a new day for search"* when Microsoft joined forces with OpenAI. Nadella, Microsoft, was challenging Google: Microsoft was powering their search engine Bing with conversational search in a chatbot supported by OpenAI.[2] The race for winning a disruption for search, as a business, was on - it was a spectacular turn of events after more than two decades of complete domination by Google. Nothing captured the *zeitgeist* better than the cover of the Economist 11th Feb 2023, as depicted in figure x.1. The battle for search was on.

2 NPR, Feb 7th, 2023: Microsoft revamps Bing search engine to use artificial intelligence

Figure P-1. The Economist nailed what AI did to search engines

And here I was, arguing that data catalogs were like search engines, just for the data in your organization. And so, back to Matt's question: How did this new technological push forward in AI change data catalogs?

At the time I was asked the question, it was too early to provide an answer. No one knew. But something was clearly going to change. I saw things that I knew from my academic background would never fly, but I also saw interesting experiments.

The first edition of my book was very well received, I was lucky to have thousands of readers all over the planet. The book was praised in reviews and featured in many podcasts, and I got to travel the world explaining my ideas at big conferences, events and for large corporations - the book even got translated.

Its success was its message. Data catalogs are engines to search *for* data, so that you could later search *in* data, directly in databases. Static, preconceived metamodels in data catalogs led to catastrophic implementations, sky-rocketing costs and no ROI - because that is basically not how a search engine should work, not on the web, not in a company.

As a PhD and an aspiring professor, I had taught knowledge organization, information retrieval and adjacent topics in my Library- and Information Science classes at the University of Copenhagen in Denmark, where I live. I went into industry, but I never forgot what I learned and taught. And it was on that basis, that I argue, that data catalogs, essentially, are like search engines. And I wrote *The Enterprise Data Catalog* from that perspective.

And then came AI. All of a sudden there was a new dimension to data cataloging that I had not covered in my book.

Why I Wrote This Book

Therefore, it is time for a second edition of *The Enterprise Data Catalog*. All the insights from the first edition are kept - it all holds true still. The AI perspective has been added on top, and it falls into two parts:

- Part I: How AI augments data catalogs
- Part II: How data catalogs (ontologies) are a source for AI

You will notice that these two parts are also the parts of this book - because we can implement and use data catalogs at scale with AI, and we can use the very structure of the data catalog, the metadata, as a source in itself, to fuel AI.

Basically, because of AI, it's also *a new day in search* for data catalogs. We are now beginning to understand what that day is about, and that is what we will unfold in the following pages. This book is about how AI augments the core message of my book, namely; *how you organize data, defines how you can search it.* And this message also has a new dimension, since the data catalog is in itself a source.

Finally, the data mesh movement was at its peak when I published the first edition of *The Enterprise Data Catalog*. At the time of publication, it was difficult to say what

would remain from the movement, after the buzz would fade. Now, that has become clear. No one talks about data mesh anymore, but two of the components in the data mesh complex stood the test of time: data products and data contracts. Therefore, this second edition covers data products and data contracts, because they are absolute key components in data catalogs.

Who Should Read This Book

This book is for everyone working in data, meaning

- Data engineering
- Data analysis
- Data science
- Data management
- Data governance

These groups of employees all work with data and they all need to know where data is, who owns data, the quality of data, how data moves, where data ends up, and who uses it. The data engineers facilitate the storage, transformation, and movement of data, and they can monitor that in the data catalog. The data analysts and scientists use the data catalog to search for interesting data - and then, once found, they will search in those data sources. Data managers and data governors use the catalog as a strategic tool to ensure that data governance policies are successfully implemented.

On top of that, a wealth of groups can benefit from using a data catalog, e.g.

- Architects
- Information security
- Data protection officers

Architects can use a data catalog for effective data migration projects, information security and data protection to ensure an empirical validation of their anticipation of what data the organization has.

Navigating This Book

This book has two parts, *Part I: The AI augmented Data Catalog*, and *Part II: Data Catalogs as a resource for AI*.

Part I: The AI Augmented Data Catalog is all about how data catalogs work and how they have improved since the first edition - mainly thanks to AI. The chapters introduce data catalogs, explains how data is organized in data catalogs and how search is

performed, furthermore how data is accessed and observed. All these aspects are improved, made easier and faster, thanks to AI. Also, we will dive deeper into data domains, data architectures, and especially focus on data products and data contracts, as these matured significantly since the first edition.

Part II: Data catalogs as a resource for AI uncovers a new perspective: Data catalogs are now becoming sources themselves, not only a catalog of sources. This new role for metadata is discussed in the light of the combination of knowledge graphs and large language models. Furthermore, the emerging standards of effective AI are explained, such as Model Context Protocol and Agent2Agent. This second ends with a future vision for data catalogs, and this is substantially updated since the first edition.

Preface for the First Edition

"This simply can't be all there is to a data catalog. What does it really do?"

About five years ago, I sat alone in the office among 20 empty desks. My company had shut off the air-conditioning to go green, so I was uncomfortably warm on top of being perplexed by the bunch of white papers, both printed and on my laptop, that were sitting in front of me. The papers explained a new technology called a data catalog. As an enterprise architect, I had been asked to implement a data catalog for our company. But first, I had to understand it.

The papers I was looking at described cool, advanced features: column-based data lineage, graph visualizations of ontologies, and workflows to access virtualized data. Useful. Mesmerizing, really. But what was the overall point of a data catalog? I was sweating, physically and mentally, trying to draw upon my experiences to figure out the potential of this new technology.

I have a BA, MA, and PhD in library and information science (LIS). I have taught LIS in university courses and been to conferences all over the world. I've seen a lot of things in this field, both good and bad. During my first job in pharma, senior management regularly called me late at night because inspections from the authorities were going haywire. The inspectors were asking them a multitude of questions: What was the temperature of this tube, in that machine, in June 1992? Where is the proof that the fermentation tank was cleaned according to the standard operating procedure (SOP)? When the data managers searched and couldn't answer, they called my team—the Records and Information Management team.

We employed our searching superpowers to find the information they needed. We were adept with our queries, used intuition and creativity to plan our moves, and drew on our knowledge to guide our search. We were able to do this because we had one guiding principle: *How you organize data defines how you can search it*. Because we knew how the data was organized, we knew ways to begin searching, modifying

how we searched, broadening, changing focus, excluding hits, and finding the information we needed. Sometimes this was easy, and sometimes it was hard, but we would get there every time.

This guiding principle has followed me throughout my career. I have cataloged furniture, weapons, human tissue, a lot of paper, and massive amounts of data. I know how to structure and operate a physical card catalog in a library. I know records and information management systems with both physical and digital storage. We both stored and cataloged data on premises, and then, later, in the cloud. Throughout everything I experienced, I saw that if you have a poorly organized data landscape, searching for the information you need will be a terrible experience. You will have to guess where to search and what to query. If your data is logically and systematically organized, however, you will know exactly where to look and what to query. It will be a much better experience.

The idea that how you organize data defines how you search for it is reflected in our web habits as well. We never really think about how we search it anymore; it's so intuitive. At work, within our company's IT landscape, it can be a different story. We search in vain—companies hardly know their own data, let alone how it is processed. Data is undiscoverable and unmanageable. If only we had an enterprise search engine…

On that hot summer day, alone in the empty office, surrounded by physical papers and dozens of open PDFs on my laptop, it suddenly hit me.

"This data catalog has the potential to become a search engine for companies! We are finally getting an engine that will be able to do for companies what search engines have done for the web. The data catalog is a search engine!"

That realization led to another a few years down the road. All of the papers I read that fateful day, along with all of the documentation that followed it, focused on explaining the complex features that are in data catalogs. They did not explain the data catalog itself and how it could revolutionize how we organize and search data. Nowhere has anyone talked about the future of the data catalog as an enterprise search engine. That realization has brought us to the book you are reading today.

Although I had the epiphany about the potential of a data catalog and it was crystal clear in my head, I was then faced with the battle of explaining the features to the important stakeholders of my company. Although I knew they would see the benefits of this tool if they only took the time to understand it, they were simply not interested, nor did they have time to study it. I had to come up with a way to reach them.

I went back to the idea of using the data catalog as an enterprise search engine. So, I asked myself, "What are people searching for? What would a data scientist be searching for? A data protection officer? A chief information security officer?"

I decided to build demonstrations of the most vital data catalog features into small stories about specific stakeholders. Each slide deck had one central picture: a minimalist search bar with the company's logo above it. I would explain the information need of a specific stakeholder, show the search in the search bar, reveal the result, then close with how the results could be used. In this way, I showed simple searches, complex searches, how to browse back and forth in the lineage of data, up and down in domains, and relationally in the graph that depicted our company. It had the same content as my previous demonstrations, but this time, it was explained from a stakeholder point of view: a specific person who was searching for something specific. And that worked.

The stakeholders not only got interested, but they also got excited. They now wanted the data catalog, because they understood that this tool was not just a collection of fancy features for data geeks. No, this tool was something way more fundamental: the data catalog could help them search and find the data they were looking for. I explained that, implemented with care, a data catalog has relevance for many of the employees in a company. This approach worked for me and my colleagues, and I hope that it will work for you and yours as well.

At the end of the day, we are all searching for something. And we search all the time. The only thing is, at work, it is very difficult to search for whatever we are trying to find. And we take that for granted, as something that we must just accept.

I'm assuming you're reading this book because you're involved with planning to implement a data catalog, improve an existing one, sunset it, or simply trying to understand what kind of technology a data catalog is: what it does, how it should be used, and if it can help you in a certain way. You might be part of the offices of the legal counsel, chief data officer, data protection officer, or chief information officer. You might be a data engineer, data scientist, or data manager, or you might be part of the data governance team. If you are, then this book will help you understand what a data catalog is and how it will enable you to find exactly what you are searching for.

However, you may also be a data catalog provider. In my book, I put forward a vision for the future of data catalogs, which you could benefit from when planning the future development of your data catalog technology.

CHAPTER 1

Introduction to Data Catalogs

A Note for Early Release Readers

With Early Release ebooks, you get books in their earliest form—the author's raw and unedited content as they write—so you can take advantage of these technologies long before the official release of these titles.

This will be the 1st chapter of the final book.

If you'd like to be actively involved in reviewing and commenting on this draft, please reach out to the editor at *shunter@oreilly.com*.

In this chapter, you'll learn how a data catalog works, who uses them, and why. Before we dive into that, we will focus on why data catalogs have become relevant in a new way for Artificial Intelligence (AI). Pay close attention to this part, as it will ensure you strategic buy-in at the executive level, also for data governance and compliance, which is usually hard to get - as you are perhaps well aware.

Then, we'll go over the core functionalities of a data catalog and how it creates an overview of the data in your organization's IT landscape. I'll learn how the data can be organized in a data catalog, and how it makes searching for your data easy. Search is often underutilized and undervalued as part of a data catalog, which is a huge detriment to data catalogs. As such, we'll talk about your data catalog as a search engine - and an AI assistant! - for your enterprise data that will unlock the potential for success.

In this chapter, you'll also learn about the benefits of a data catalog in an organization: a data catalog improves data discoverability, subsequently ensuring data governance and enhancing data-driven innovation. Moreover, you'll learn about how to set

up a data discovery team and you'll learn who the users of your data catalog are. I'll wrap up this chapter by explaining the roles and responsibilities in the data catalog.

OK, off we go.

The AI Data Catalog and as Source for AI

Before we jump into the nuts and bolts of what a data catalog is, let's take a moment and focus on why data catalogs have become relevant both

- with AI, and
- for AI.

With AI. Data catalogs are now substantially easier to implement, use and scale, thanks to many of the features in them being supported by AI. This will become clear already in this chapter, in the search examples below, and in we will go deeper into this in the other chapters of *Part I*.

For AI. As you will also see in this chapter, data catalogs are now part of a bigger search infrastructure, via AI assistants. AI assistants not only search on data catalogs but in all sources connected to the AI assistant. This has created a remarkable shift: The data catalog is not only a tool leading to sources, it has become a source in itself.

You might know the term *AI assistant* under the term *chatbot*. Think Claude, Mistral, OpenAI and others, only for dedicated enterprise use. Throughout this book, we use AI assistants.

Furthermore, data catalogs - if they are built on knowledge graphs - are a rich source for AI. Graphs have proved to increase precision greatly when applying Large Language Models (LLM) for generative AI projects. Furthermore, agentic architectures are also executing tasks more effectively when supplemented by a knowledge graph.[1] Overall, the combination of AI and knowledge graphs is promising and we will discuss this in detail in *Part II*.

You may think, If you are in data governance or data engineering: "Why does all this AI stuff matter to me?" I need my data catalog anyway. The answer is simple. It matters to you because this evolution has made it significantly easier to get your enterprise decision makers on board in implementing a data catalog, and you can also expect a smoother experience in rolling it out for new end users.

1 There will be references to relevant scientific articles documenting this, in Part II also.

The Core Functionality of a Data Catalog

At its core, a data catalog is an organized inventory of the data in your company. That's it.

The data catalog provides an overview at a metadata level only, and thus no actual data values are exposed. This is the great advantage of a data catalog: you can let everyone see everything without fear of exposing confidential or sensitive data. In Figure 1-1, you can see a high-level description of a data catalog.

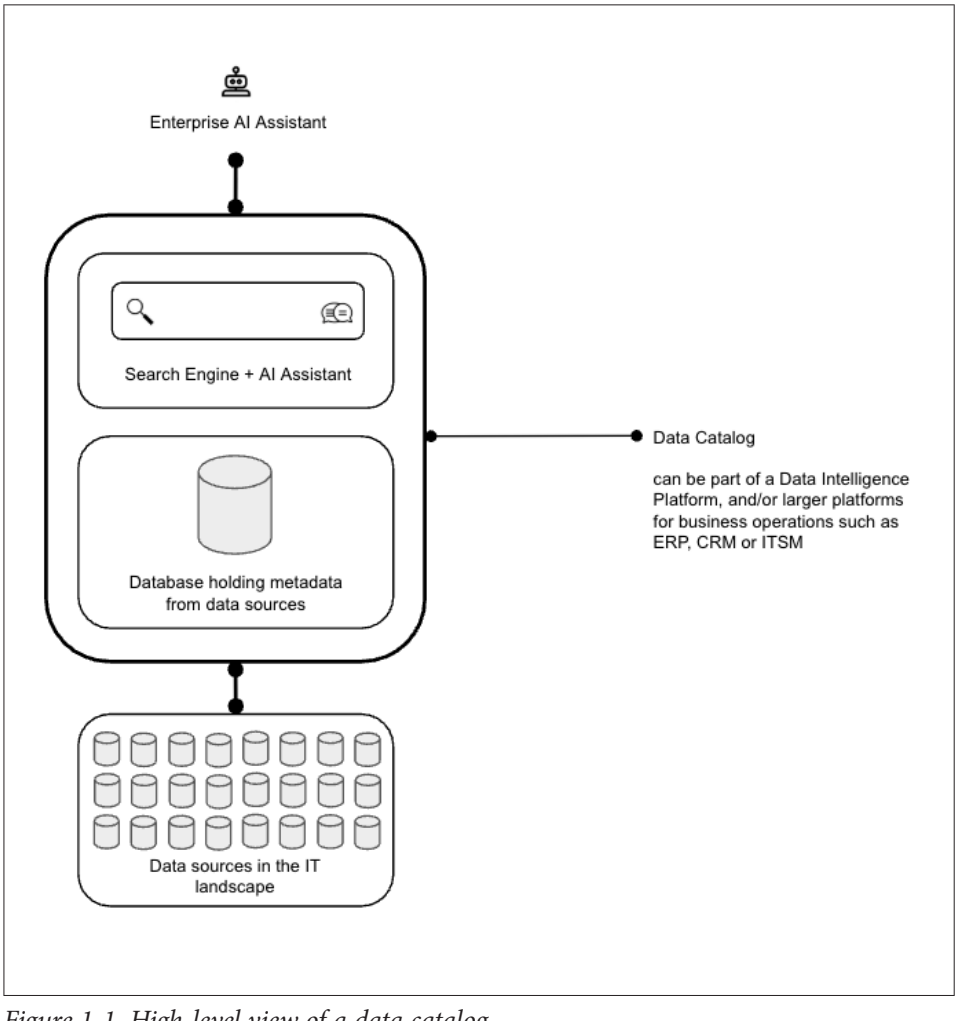


Figure 1-1. High-level view of a data catalog

A data catalog is basically a database with metadata that has been extracted from data sources in the IT landscape of a given company. The data catalog also has a search engine and an AI assistant inside it that allows you to search the metadata collected from the data sources. A data catalog will almost always have a lot more features, but Figure 1-1 illustrates the necessary core components. And in this book, I argue that the search capability is the single most important feature of data catalogs.

In this section, we will discuss the three key features of the data catalog, namely that it creates an overview of the data in your IT landscape, it organizes your data, and it allows you to search your data. Let's take a brief look at how data catalogs do this.

With a data catalog, your entire organization is given the ability to see the data it has. Used correctly, that transparency can be very useful. For example, data scientists will no longer spend half their time searching for data, and they will have a much better overview of data that can really deliver value. Imagine the possibilities. They could be using their newfound time to analyze that data and discover insights that could lead the enterprise to developing better products!

Create an Overview of the Data in the IT Landscape

Creating an overview of the data in your IT landscape involves finding and displaying all the data sources in it, along with listing the people or roles attached to it.

A data catalog can pull metadata with a connector that scans your IT landscape - each data source will have its own connector. Furthermore, as data product architectures are becoming more frequent and mature, data products can automatically publish themselves at the metadata layer, to the data catalog. We will discuss data products and data contracts in depth in Chapter 8.

The IT landscape that is reflected in your data catalog will get business terminology added to it terms that are created in the data catalog (or inherited from other metadata repositories) and organized in glossaries. We will discuss glossary terms in Chapter 2 and how to search with them in Chapter 3. Besides glossary terms, you can also enhance your data catalog's assets with other types of metadata, complete with additional descriptions, classifications, and more.

Furthermore, a data catalog has various roles and permissions built in, such as data steward, data owner (data catalogs have different role type names), and other roles that all carry out specific tasks in the data catalog. I will describe those roles for you at the end of this chapter.

Once your enterprise data is represented and described in your IT landscape and assigned selected terms, other metadata, and roles to it, it's searchable in the catalog.

You Will Always Be Able to See More Metadata than Data

No employee can see all the data in the IT landscape. Even more confusing: no employee can see what data others can see. Basically, no one knows about all the data in the IT landscape: it's opaque.[2] This reality is also referred to as *data silos*. *Data silos* emerge when several groups of employees work with their own data in their own systems, isolated and unaware of the data in the rest of the organization.

This state—the data siloed state—is the root cause of an immense set of problems in many organizations, which the data catalog addresses and ultimately solves. These problems include data analytics applied to data lacking quality, incomplete datasets, and data missing security and sensitivity labels.

This perspective can also be flipped: data silos are connected, but no one can see it or knows how they are connected. This makes the data siloed state even more dangerous, but as you will see, capabilities in the data catalog can help map the data.

In the data catalog, it's the complete opposite situation of the IT landscape itself. Everything in the data catalog is visible to all employees. Everyone can see everything —at the metadata level. And accordingly, all employees can get an idea of all the data in their company, based on that metadata. They are mindful and aware of the data outside their own, now past, data silo.

The more the data catalog expands, the more everyone can see. If this makes you think that a data catalog holds remarkable potential, you're not wrong—and you will discover the magnitude of that potential in this book.

Based on my experience, I suggest you organize data in a data catalog in the following way.

Organize Data

As a data catalog crawls the IT landscape, it organizes the metadata for data entities within the landscape as assets pertaining to a data source and stores them in domains. The same goes for data products that are published to the data catalog. Data - put simply - has to belong somewhere.

Therefore, you play a big part in this: you must design the domains and part of the metadata that data is to be assigned. And keep in mind that most data catalogs offer automation of this process—it should not be a completely manual task to add metadata.

2 certain cyber security functions may be able to see "everything" in the IT landscape.

What is a data asset? An asset is an entity of data that exists in your IT landscape. It could be a file, folder, or table, stored in a data source such as an application or database, etc. Assets are, for example, documents in a data lake, SQL tables in a database, and so on. When the data catalog collects metadata about the asset, whether by push or pull methods, it obtains information such as the asset's name, creation date, owner, column name, schema name, filename, and folder structure. Overall, the collected metadata depends on the data source and the data that sits in it. You must add metadata to the asset beyond what was populated by automatically. We'll talk more about this in Chapter 5.

And so what is a data source? Simply put, a data source refers to where the data that is being exposed at a metadata level in the data catalog comes from. It can be an IT system, application, or platform, but it can also be a spreadsheet. In the context of this book, the type of data source is irrelevant because all data sources can be treated in the same way.

Finally, what is a data product? We will discuss this in chapter 8, for now, keep in mind that a data product is an autonomous, consumable data asset, in the sense that *it has been made* consumable. Contrary to an asset, that simply exposes that a particular type of data exists, somewhere.

A domain is a group of data assets and or data products that logically belong together. These assets may stem from one or more data sources. For example, a domain with finance data may both have analytics data sources and budget data sources. It is critical to define your domains with care because they should be intuitive for employees outside that domain—and they should be intriguing to explore for those employees—a data catalog is an initial step toward breaking data silos!

So far, data catalogs have only been described in the data-management literature. In that literature, the understanding of domains typically refers to domain-driven design (DDD), as an attempt to push DDD thinking into data. In this book, you'll find domain architecture based on the century-long tradition of domain studies in information science. This will provide you with a deeper, more functional understanding of domains than in normal data-management literature—you'll find all this in Chapter 2.

Now that you have a better idea of how assets, data sources, data products, and data domains work, let's look at a few examples of how they all fit together. Figure 1-2 shows a table in a database (also called as a data source, in a data catalog) and how the table visible in the data catalog.

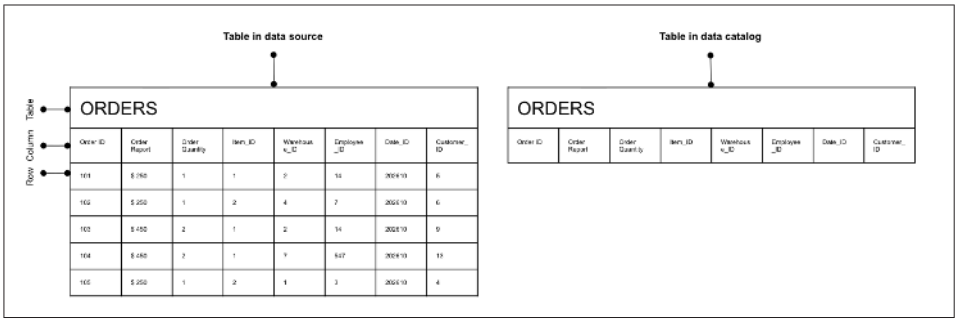


Figure 1-2. Table in a data source and how it's visible in the data catalog

As you can see to the right side of the figure, no values are included in the data catalog. In this case, sensitive data `customer_ID` is not visible in the data catalog as it is in the data source.[3] In the data catalog, only the column name is displayed. In this way, everyone can see everything in the data catalog. It's the actual values in, for example, tables that makes it impossible from a governance perspective to create a complete overview of data in your company (and then, there is the obvious technical boundary of such a scenario). With the data catalog, those days are over - you can discover more and more data.

Column names and other metadata visible in the data catalog can also contain sensitive or confidential data. Methods must be in place to make sure that such metadata is not visible to the users of the data catalog.

You can add metadata to what you represent in a data catalog—in this case, a table—both at the table level and for each column. Every piece of metadata added will inscribe it with context relevant to the knowledge universe of your organization. This will make your asset more searchable. We'll talk more about how to organize it in Chapter 2 and how to search for it in Chapter 3.

Furthermore, it's important to understand that data is organized into vertical, horizontal, and relational structures, as can be seen in Figure 1-3 and more exhaustively in the figures in Chapter 2.

Vertical organization enables you to pinpoint exactly from what domain in your organization the data belongs to. This is achieved through domains and subdomains. In the `Orders` table in Figure 1-3, the vertical organization specifies which part of the company the data comes from; for example, finance.

3 reference til Serras bog

The horizontal organization of assets allows you to display how the asset moves in your IT landscape. This is done with *data lineage*. Data lineage depicts how data travels from system to system and, ideally, how the data is transformed as it travels. Data lineage can be subdivided into many different layers, in Figure 1-3 below, you see the layers being divided into

In the `Orders` in Figure 1-3, lineage would, for example, display that the data resides in a database and that it is used in a Business Intelligence (BI) report, indicated by an arrow to the right of the asset, pointing toward the BI report.

The curved lines organize how certain types of data relate to other types of data and, if done correctly, can render these relations in a *knowledge graph* - and this is the key to powerfully organize and search your data. In the `Orders` table in Figure 1-3, the relational organization of the `Date_ID` column could, for example, be related to other temporal metrics in other data sources, e.g., from manufacture data, referring to naming conventions for batch series productions, and so on.

All together, an organized table in a data catalog is depicted in Figure> 1-3.

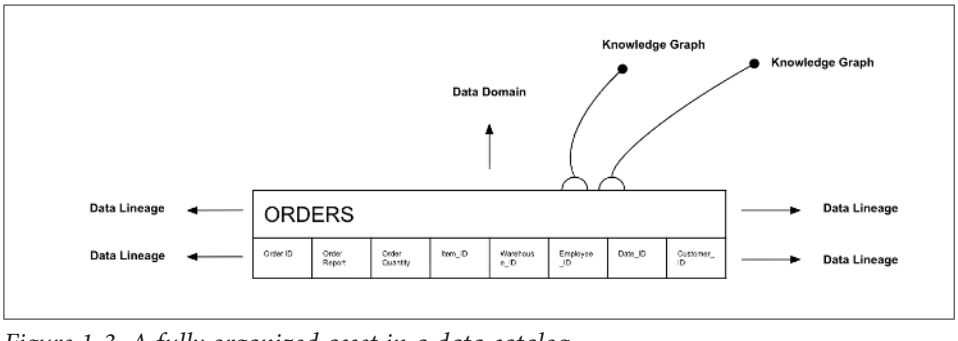


Figure 1-3. A fully organized asset in a data catalog

Once your assets have been organized into neat vertical(domains), horizontal(data lineage), and relational(knowledge graph) structures (for examples of this, check out Chapter 2), you might be tempted to think that your job is done and you no longer need to work on your magical data catalog. That is not the case! You should not consider a data catalog to be a repository that only needs to be organized once. You should always be open to reorganizing assets and improving the metadata quality and coverage. Not only will it ensure things are neat and tidy, but it will optimize your data catalog for search.

Accordingly, let's take a first look at searching a data catalog.

Enable Search of Company Data

Search is one of the key functionalities of a data catalog. It is often treated as just a feature, but it can be so much more than that if you make it the driving factor of your data catalog strategy. Think of your data catalog as a search engine and an AI assistant, the same kind that you'd use to search and ask questions to the web.

A data catalog and a web search engine are similar in that they both crawl and index their landscapes and allow you to search that landscape. The main difference is that while a web search engine covers the web as a landscape, a data catalog covers your organization's data in the IT landscape.

A data catalog is also like an AI assistant, in the sense that you can interact with it in natural language and have a conversation with it. It is important for you to remember, that this feature gives your data catalog a certain plasticity - it can bring about more search options in it, and it also sometimes makes you search the data catalog outside of the data catalog itself - the data catalog can be a source for searching with an Enterprise wide AI assistant. You can see an example of this in Figure 1.**X** in this chapter.

So, what does it look like when you treat your data catalog as a search engine and AI assistant? Let's take a look at one in action.

Throughout this book, we'll be looking at the data catalog of Hugin & Munin. Hugin & Munin is a fictitious Scandinavian architecture company that specializes in sustainable construction that uses wood from forests close to their building sites.

The Hugin & Munin data catalog revolves around organizing data and searching for it. Figure 1-4 shows the interface for the Hugin & Munin catalog. The search bar allows you to enter terms to do a regular search of the data catalog, but you can click the little squirrel underneath to enter directly in conversational AI search. The magnifying glass allows you to use the browse function and a pile of books icon gives you access to the glossaries. Note that this looks very similar to most popular web search engines and AI assistants.

Figure 1-4. The data catalog frontend in Hugin & Munin

Let's look at how you might use this data catalog. Say that you are an employee at Hugin & Munin and you overhear a group of people having a conversation during lunch. They talk about this clever data scientist named Kris, mentioning that he's the owner of some visionary data products in your company's data catalog (you'll learn about data owners later in this chapter; right now it's not important). Such data products could be useful in the project you are currently working on. Before you can ask the group about how to contact Kris, they rush to a meeting room they forgot all about. Back at your desk, you search the data catalog as depicted in Figure 1-5.

Figure 1-5. Advanced search for the exact information you need

In earlier days, you would have had to type keywords and complicated operators to make this work, something like this:

```
Data Owner: "Kris*" AND business glossary: "Data Science" And Asset Type: "Data Product"
```

But luckily, thanks to the recent developments in AI, those days are over. The AI features of a modern data catalog will automatically extract the relevant keywords and

boolean operators in the natural language you type, translate your questions into a structured query statement, and then deliver you the results.

Search results in the data catalog for this search: You find seven splendid data products by a data scientist called Kris Kegelstrom. They are data about the long term durability of the facades of Hugin & Munin family houses. You are certain these are the data products your colleagues talked about at lunch.

However, even though the description of the data products are very informative, you feel you lack some more context about what the data products can be used for. Basically that conversation between your colleagues discussed some really cool ideas and perspectives going in many different directions, and that's not really completely captured in the description of the data products in the catalog. Of course you will find your colleagues and ask them. But before you do that, you get an idea. What if you use your enterprise wide AI assistant - called the Hugin & Munin chat - to find the seven data products in the data catalog, and then check if the data products have been discussed in some of the many, many threads in slack?

This is possible because both data from the data catalog and data from slack has been made available on MCP servers, making it queryable for the Hugin & Munin AI assistant. Your prompt is displayed in Figure 1-6.

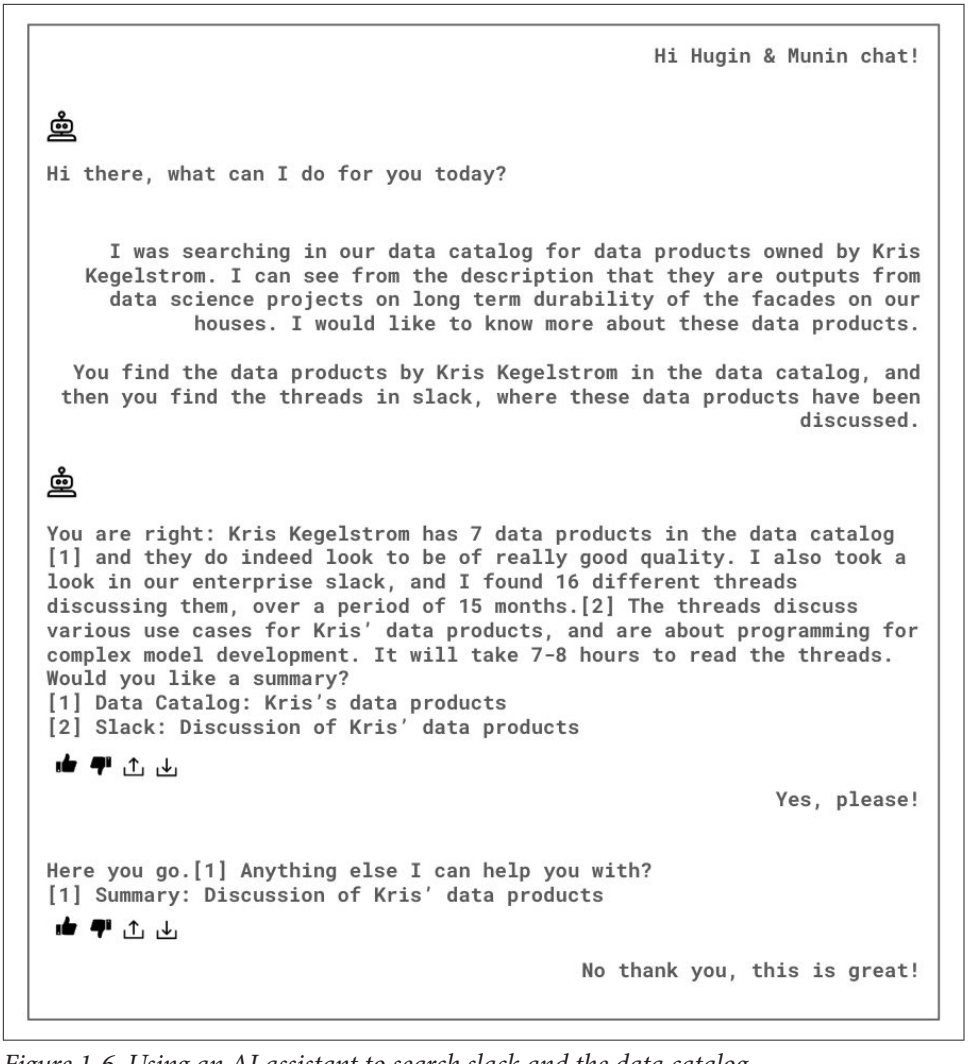


Figure 1-6. Using an AI assistant to search slack and the data catalog

Basically, your assumption was right. Kris' data products have indeed been the topic of discussion in slack - 16 different threads have discussed them, for over a year in total. The summary by the AI assistant of the threads makes it more easy for you to move forward with your own ideas.

You've got a glimpse of how search works in this example, and just how powerful a data catalog can be in achieving great things with data. And we have only begun, the search is described in full depth in Chapter 3. Remember - the more searchable your data is, the more you enable the one big benefit of a data catalog: data discovery.

Data Discovery

A data catalog enables all employees to search all data in their company. Searching and actually finding data is called *data discovery*, and that's what a data catalog is all about.

Nevertheless, data discovery is rarely thought of as searching *for* data, but often as searching *in* data, in databases, to find new insights about customers, products, etc.

Searching *for* data can be haphazard conversations with colleagues, by memory, or it can be structured, meaning that searching for data takes place in a formalized manner in a solution designed for the purpose of searching for data, for example, a data catalog.[4] The difference between searching for data and searching in data may strike you as not very important—but it is! And we will discuss it in detail in Chapter 3.

Put simply, *data discovery* begins with discovering that certain data exists at all, not what's inside it. Once you get your data catalog up and running, you will exponentially accelerate data discovery *in* data, because the preceding search *for* data is remarkably more effective with a data catalog than without it.

Data discovery *for* data, in a data catalog, has a distinct target state: *ambient findability*. This term was coined by Peter Morville in the first literature that shed an intellectual light on the powerful search engines on the web that arose in 1995–2005:

> Ambient findability describes a fast emerging world where we can find anyone or anything from anywhere at anytime.[5]

Today, data catalogs are emerging as the equivalent of web search engines and AI assistants, only for the data in your enterprise. And data catalogs, too, should strive for ambient findability. That's how smooth data discovery *for* data must be: in your data catalog, you should be able to find anyone or anything from anywhere at any time—in your enterprise.

Ambient findability is completely unrelated to how you search *in* data. Searching in data is so persnickety and subtle that an entire field has evolved out of it: data science. I discuss this more in Chapter 3.

Data discovery in a data catalog serves several purposes:

4 G. G. Chowdhury, *Introduction to Modern Information Retrieval* (New York: Neal-Schuman Publishers, 2010), chaps. 1 and 2.

5 Peter Morville, *Ambient Findability: What We Find Changes Who We Become* (Sebastopol, CA: O'Reilly, 2005), 6.

- Data engineering
- Data analytics
- Data governance, risk & compliance
- As a source for AI generative search and agentic orchestration

Data engineering is an activity that will use the data catalog for monitoring of data pipelines, through technical performance metrics known as *data observability*. Also, the data quality metrics of data will be monitored by data engineers (although data architects will have the subject matter expertise of defining the quality of data).

Data analytics supported by a data catalog is pretty simple: data scientists—analysts and similar profiles—all need data. Without a unified, global overview of data in your company, these highly paid employees just work with the data they happen to know—in their data silo—and not the best-fit data for what they want to do. You can change that with a data catalog and create a complete overview of all the data in your company. This means that data-driven innovation can accelerate and deliver substantially more value.

Data governance, risk and compliance supported by a data catalog has many advantages, and I'll discuss these in depth in Chapter 4. The most important one is the capability to classify all data in your IT landscape both in terms of sensitivity and confidentiality. This will be of great value for your data protection officer (DPO) and your chief information security officer (CISO)—indeed, for your entire company. A data catalog applies rules to its pull/push capability so that all its assets are automatically assigned a sensitivity classification and a confidentiality classification. You can take a look in Chapter 2 about this for more details. For now, just remember that the power of automated classification of sensitivity and confidentiality directly on your IT landscape is a bedazzling feature that won't be difficult to sell.

As a source for AI generative search and agentic orchestration. AI architects and AI engineers will use a data catalog as a source for deeper, enterprise wide search, and for orchestration for an agentic architecture/mesh. This is a new, promising use case for data catalogs, but it requires that they are based on knowledge graphs. Only a knowledge graph powered data catalog can deliver full potential for AI, because it provides a useful semantic structure, that guides AI, and help it take better actions.

Altogether, these purposes can be grouped into one term, *data intelligence*, a term we will discuss more in chapter 5.

Data catalogs are also used by people who do not have many tech skills; I discuss them in the following as everyday end users.

The Data Discovery Team

A data management job—including managing a data catalog—is not the job of one person alone. Rather, it is the work of an entire team to implement, maintain, and promote the usage of the data catalog across your organization. Although you could call this your data catalog team, I encourage you to call this your data discovery team instead. This tells everyone not just what technology you use, but on what capability you deliver, which is data discovery.

Data discovery teams can focus solely on data catalogs or more widely on all metadata repositories. You should push for the latter: preferably, the data discovery team owns and curates all metadata repositories like the CMDB (configuration management database), data sharing agreement system, etc. that describe everything within the IT landscape. In this way, it can promote data discovery from the totality of sources where these are exposed at a metadata level. I discuss the Data Discovery team in depth in my book *Fundamentals of Metadata Management* (O'Reilly, 2025)

The Data Discovery team maintains the high level overview of the data catalog, called the *metamodel*. You can see an example of a metamodel in Figure 1-9. The metamodel is the model that provides an overview of all types of entities in the data catalog. The metamodel also includes all relations between the entities. For example, departments have people, perform processes, and are supported by technology. Basically, the metamodel defines how you can physically structure your data catalog, based on conceptual metadata structures.

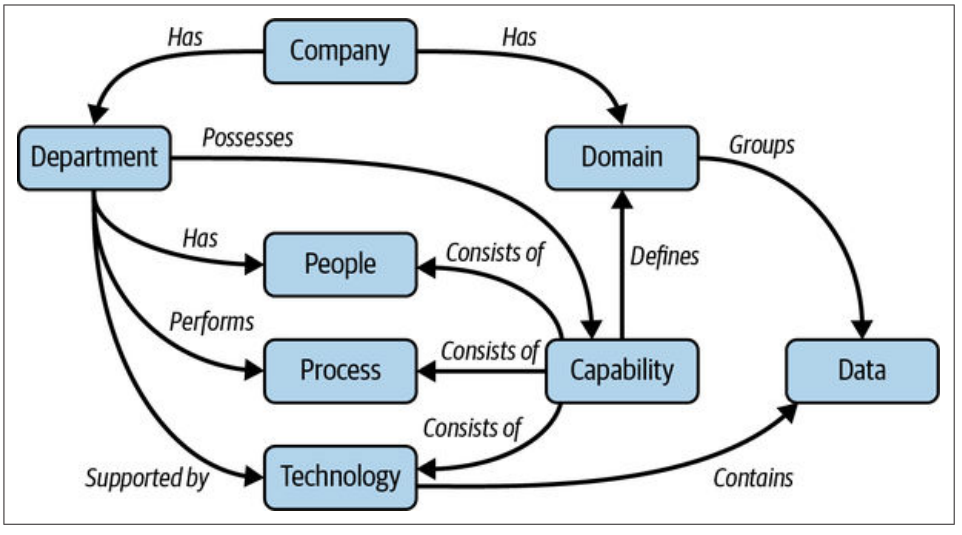


Figure 1-7. Example of a metamodel in a data catalog

Consider the metamodel in Figure 1-9. In this hypothetical example, a *company* has two entities, *departments* and *domains*. Departments and domains are not alike, as we will discuss in Chapter 2. A *department* has *people*, performs a *process*, and is supported by *technology*. Furthermore, a *department* possesses a *capability. Capability* defines a *domain*, and *domain* groups the *data* that the *technology* contains.

At a first glance, a metamodel may provoke dizziness. But the metamodel is there to provide the best possible structure for the data that is represented in the data catalog. It organizes data into its most relevant dimensions so that it is as easy to search for as possible.

Knowledge graph–based data catalogs have flexible metamodels. The metamodel in such data catalogs can be visualized, expanded, and searched without limits. Beyond purely technical, raw basic data catalogs, knowledge graph powered data catalogs have won the market, as they very powerfully caters to the most important feature in a data catalog: search.

Flexible metamodels provided graphs is a big topic, and it will only become more important with AI. We will cover it at length in Chapter 2, 5, and 6.

Data Catalog Ownership

Where should ownership of the data catalog be placed? There is not one, but several possibilities:

- Data governance
- Chief data officer (CDO)
- Data analytics

The benefits of each way can be described as follows:

Data governance ensures that data is managed in accordance with regulations and standards. It also focuses on data quality aspects, ownership, etc. The advantage of placing the data discovery team in a data governance part of the company is that it leads to better data compliance and efficiency of the operational backbone. You will ensure that confidential and sensitive data is protected. Nevertheless, if such an approach is used, a data catalog should merely be considered an expense to ensure data governance, and not as the key component it is intended to be for data-driven innovation.

Having a *CDO* responsible for the data discovery team is the ideal, but also a rare setup for a data catalog. In this case, the data discovery team is a staff function for the CDO. The CDO writes and puts into action the executive data strategy of a company and should therefore have a full overview of all data at hand. In such a case, the executive data strategy is based on empirical facts, and outcomes are measurable.

Placing the data catalog in a *data analytics* business unit puts the data catalog directly into action where it delivers the most value: innovation. However, the risk of this setup is a lack of control. Without firm data governance, the data catalog can risk exposing confidential data or processing sensitive data in a way that is a liability to your company or in a way data subjects have not consented to. It can also create difficulties for data quality, which is a time-consuming effort that an energized team seeking results could be tempted to neglect.

End-User Roles and Responsibilities

End users of a data catalog fall into three categories:

- Data analytics & AI end users
- Governance end users
- Everyday (efficiency) end users

Data analytics and AI end users search the data catalog for data sources for innovation, and their data discovery does not end in the data catalog when they search for data. Data discovery *for* data leads to data discovery and data exploration *in* data, as we will discuss in Chapter 3. Data analytics end users should be considered the most important end users of the catalog, as they will deliver the return on investment (ROI) for the data catalog. They do so by innovating new offerings to customers,

based on data they have searched, found, analyzed, and used for business opportunities and growth.

Governance end users primarily search the data catalog for either confidential data or sensitive data—or both—in order to protect that data. They do so both as the catalog expands with new data sources (I discuss this in Chapter 5) and on an ongoing basis, when performing risk assessments and during daily operations. They also use the data catalog to get a more managed approach to who can see what data in the organization. The data catalog will enable them to increase the data governance of the company, but an ROI is more difficult to document in comparison with data analytics end users.

Everyday end users are likely to become the most substantial group of end users in the future. You can go to Chapter 8 to check what that future looks like in detail. At the point where the data catalog truly evolves to become a company search engine, employees are going to use it for everyday information needs. These are expressed with simple searches and are aimed at reports, strategy papers, SOPs, and basic access to systems. Currently, everyday end users of a data catalog are not a very big group. But you can plan your implementation in such a way that everyday end users become larger in numbers, with the effect that the data catalog gets more traction in your company. I discuss this in Chapter 5.

All end users have one or more of the following roles and responsibilities in the data catalog:

Data source owner
: The data source owner is also known as simply the system owner or data custodian in traditional data management.

Domain owner
: A domain owner manages a specific collection of assets. The domain owner ultimately defines which assets belong in the domain and who should have the different roles in the domain.

Domain steward
: A domain steward takes on more practical tasks such as conducting interviews with upcoming data source owners, managing the domain architecture, and providing access to data.

Asset owner
: The asset owner is the owner of the data in the data source. Typically, data ownership spans multiple data sources (as data ownership spans multiple systems), and it can also in rare cases span multiple domains. It is the asset owner that grants access to data upon request.

Asset steward
: An asset steward has expertise on a particular subset of assets (an entire data source or parts of data sources) in a domain.

Term owner
: Term owners typically own a large subpart of glossaries related to one or more domains in the data catalog.

Term steward
: Term stewards are responsible for managing term lifecycles. (See Chapter 7 for details.)

Everyday end user
: Everyday end users are able to search the data catalog and request data from asset owners.

Collectively, the end users of a data catalog constitute a social network. If they can work in groups independently of the data discovery team, the data catalog will provide the most value. See Chapter 5 for details on this.

Summary

You have now gotten the first impression of a data catalog. This unique tool represents a powerful step for your company toward better, more secure use of your data.

Here are the key takeaways of the chapter:

- Data catalogs are increasingly getting powered by AI, and this is a huge benefit for the end user - organizing, searching, and accessing data is becoming more simple.
- AI is also changing the nature of data catalogs: A data catalog is not only a tool to find sources, it is also becoming a source in itself.
- Data catalogs are organized in domains that contain data assets and data products. These are metadata representations of data in source systems.
- Data sources have either been crawled by the data catalog connectors or in the case of data products, been published to the data catalog.
- AI has now made it possible for your data catalog to cater for a completely free and flexible search in natural language, however complicated the topic.
- You can also search the content of your data catalog through your enterprise AI assistant, and combine it with other sources.

- The strategic benefit of a data catalog is data discovery. For the first time, companies are now able to discover all their data in a structured and endless way.
- Data discovery serves data-driven innovation and data governance. Innovation is the most important and is the reason why data catalogs emerged in the first place. Data governance, on the other hand, is not as profitable but is important in its own right—it secures data.
- Accordingly, end user types fall into categories of data analytics, governance, and everyday users. The end users can have different and even multiple roles and responsibilities in the data catalog.
- There are three possible setups for data catalog ownership:
  - The team can be focused on data governance, with the risk of losing the innovative potential of the data catalog.
  - The team can be focused on innovation, with the risk of compromising data governance.
  - The best possible setup is as a staff function for a CDO, who should take every strategic decision based on the data that's actually in the company, be it for innovative or governance purposes.
- Data catalogs powered by knowledge graphs have won the market, and there are two reasons for that:
  - A knowledge graph powered data catalog enables a flexible metamodel, that allow you to organize, visualize and search the data catalog with great impact
  - A knowledge graph powered data catalog is a unique source for AI, that will increase output of generative technologies and agentic decisions.

In the next chapter, we'll talk about how you organize data in the data catalog.

CHAPTER 2

Organize Data: Design a Robust Architecture for Search

A Note for Early Release Readers

With Early Release ebooks, you get books in their earliest form—the author's raw and unedited content as they write—so you can take advantage of these technologies long before the official release of these titles.

This will be the 2nd chapter of the final book.

If you'd like to be actively involved in reviewing and commenting on this draft, please reach out to the editor at *shunter@oreilly.com*.

You can't discover and map the entire world in a day. Likewise, you can't discover and map the IT landscape of your organization in one go. As you discovered in Chapter 1, the effectiveness of a data catalog is heavily dependent on how well it is structured and managed—this impacts how well it can be searched. Although it may sound straightforward, organizing data with metadata isn't simple. You will have to ask yourself: what is the most logical way to group data? What's the most relevant metadata for my data? How do my data relate? Can they relate in multiple ways? And what is the interplay between how confidential data is, and how sensitive it is?

In this chapter, we'll go through these kinds of questions and walk through the process of gathering and organizing the data in a data catalog. We'll begin with how to organize domains, proceed to a brief discussion of how you populate the domains with data, and end with how to organize your data once it's represented in the data catalog.

Let's first have a look at how AI is augmenting the way you organize data.

Using AI to Organize Data

Using AI for enterprise tasks is really great. In fact, it's a gamechanger! Many of the manual, repetitive tasks are done for you. For data catalogs in particular, there are great benefits of the AI features that are becoming more and more frequent, and very specifically for organizing data, AI is simply a catalyst for success. Implementation of data catalogs is not a given - that is why you are reading this book, after all - and with AI, your chances of successful implementation is significantly increased. The potential is reduced manual labor: avoid repetitive tagging, avoid enhancing descriptions, etc.

However, there are also pitfalls. Take great care in avoiding tautologies and false ontologies:

Tautologies
: Certain tasks may seem tempting to do with AI, but can be tautological, meaning, they just repeat metadata without adding value, or, even worse, creating noisy search results. Let's take a simple example, a column name in a table, named `Customer`. Adding the description in a data catalog by the help of AI that states *"this table with a column name that describes a customer"* provides a seemingly nice human readable context. But in fact, this is nothing but an obstacle for powerfully searching for your data. Because humanly added metadata must serve the purpose of (positive) discrimination - that we make all potential search hits stand out from one another. That's not done by repeating metadata with no discriminatory potential. We will dive more into the mathematics of this in chapter 3. However, remember that you're always organizing your data so that you can search for it. AI is sometimes a tricky friend in that regard.

False ontologies
: Another thing that you cannot rely on, is an LLM powered generation of your core ontology, your metamodel - materialized in knowledge graph. You will sometimes see such tempting technological offers from various software vendors. The idea is, that these companies scan a critical mass of text in your enterprise, SOP's, emails, slack threads, and, by relying on AI, generate a knowledge graph out of the input from the scan, that could potentially work as your metamodel. Take great care in not following that advice - metamodels based on knowledge graphs for data catalogs usually are small, and extremely precious. Take the relatively short time it takes to craft them by hand, the return is enormous - and an accelerator for good AI use cases - as you will learn below in this chapter.

However, organizing data with AI at scale is a real thing - it's a gamechanger. So, with the above perspectives at hand, think of the potential of AI as you read this chapter, and when you are working with organizing data in your data catalog.

Organizing Domains in the Data Catalog

As I discussed in Chapter 1, a domain groups data that logically belong together. Accordingly, the first thing you need to do is to create your domains. You do not need to create them all at once, just the ones you need to represent in the catalog. However - they must follow the same methodology, so pay close attention to that. But what does organizing the domains really mean in the context of a data catalog? In a data catalog, it is up to the domain owners to define what data belongs in their domain. In the following subsections, I will provide you with a guide to architect domains.

Domain Architecture in a Data Catalog

The task of organizing your domains can be a messy one. Without a reference architecture for your domains, you might not even know where to start.

Imagine a standard classification tree, like the one you can see in Figure 2-1. It has a root that leads to separated subcategories, which again can have further subcategories and so on. It's a structure that can expand as needed in breadth and depth.

The top level that you see in Figure 2-1 is the value chain of your enterprise. The absolutely most high level depiction of the value that is being delivered by the organization you find yourself in. All enterprises - public, private, NGO's - have a value chain. Find it, to ignite your organization of data in your data catalog!

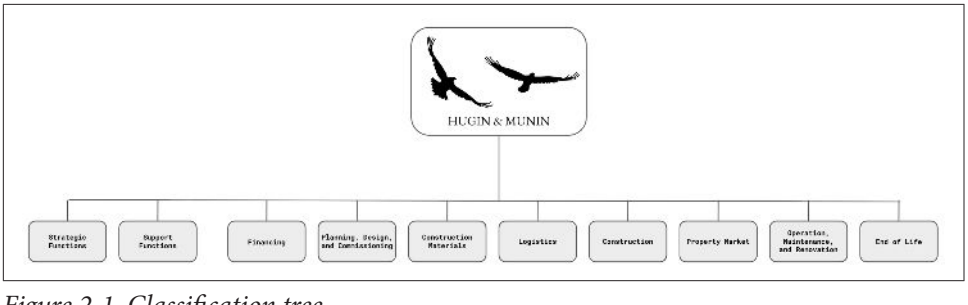


Figure 2-1. Classification tree

That point of departure is how you should organize the domains. But *exactly* how?

Have no fear. I have created a domain architecture for data catalogs to get you started and group your data sources in a logical, systematic way. You can see it in Figure 2-2. With this architecture in hand, you get a framework that will allow you to keep a firm hand on the tiller when you organize your domains. This is the hands-on guide to group all your data sources in a logical, systematic way.

The top level of Figure 2-2 is the data catalog main entry. The *main entry* is the root of all the domains and a logical starting point. Think of it as your entire catalog:

everything is subdivided from here. It's not recommended (and in many data catalogs impossible) to have multiple main entries, for two reasons:

- All data from each data source can be stored in only one place. This means that all levels need to relate to one top level, as it all constitutes subdivisions of one body of data: the data of your organization.
- Your data discovery team needs complete control of the entire data catalog, as I discussed in Chapter 1. Therefore, you need one top level, from which all roles and responsibilities are assigned in the lower levels.

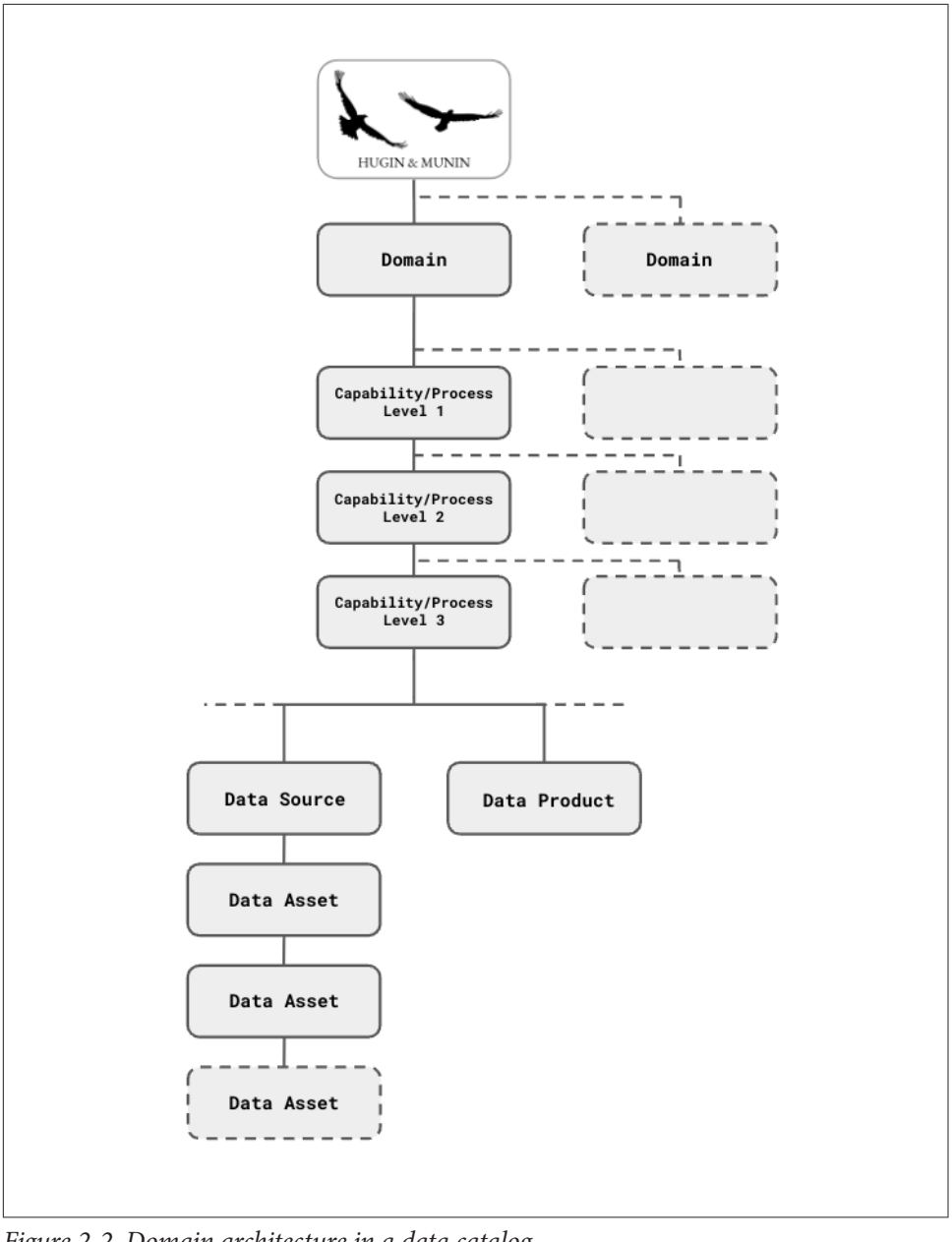


Figure 2-2. Domain architecture in a data catalog

Before I talk about the various layers of the data catalog architecture, you first need to have a deeper understanding of what is meant by *domain*.

Understanding Domains

If you want to understand what a domain is, two fields can deliver answers:

- Domain-driven design
- Library and information science

It's domain-driven design (DDD) that has been dominant in data management—and subsequently most known in a data catalog context. But since the first edition of *The Enterprise Data Catalog* in 2023, Library and Information Science methodologies and practices have won substantial influence within Data & AI. And, as you will see later in this subsection, DDD is not a completely proper fit when you architect domains in the data catalog. Therefore, I will first briefly describe the understanding of a *domain* in DDD and information science.

DDD emerged in the early 2000s and was formulated in the book *Domain-Driven Design* by Eric Evans.[1] DDD enables software engineers to better understand and cater to the context to create usable, logical software. In DDD, software design is driven by the domain it's created for—domain-driven design.

Remember: DDD is intended for the creation of software. You can see just how intermingled domain and software is in DDD, in this quote from Eric Evans:

> Every software program relates to some activity or interest of its user. That subject area to which the user applies the program is the *domain* of the software.... To create software that is valuably involved in users' activities, a development team must bring to bear a body of knowledge related to those activities.... Models are tools for grappling with this….[2]

So, in DDD, understanding a domain is about creating software that models users' activities or interests—taking into account the knowledge those activities/interests rely on.

These days, DDD is becoming relevant for a new purpose, this time not for software, but for data. It is the movement known as *data mesh*, which has applied DDD for data. Data mesh suggests a way to create scalable data infrastructure that allows for analytical data to spread faster, and easier, in a federated governance model where each business unit is responsible for storing, exposing, and providing access to its data. I discuss data mesh architecture in more depth in Chapter 6, as well as how I see this architecture in relation to a data catalog. For now, you only need to know that data mesh understands domains as defined in DDD.

1 Eric Evans, *Domain-Driven Design: Tackling Complexity in the Heart of Software* (Upper Saddle River, NJ: Addison-Wesley, 2003).

2 Ibid., Part I.

And now, we are at the core of the problem: DDD was not intended for data architecture, but for the creation of software. Thought leaders in the data mesh movement such as Zhamak Dehghani and Piethein Strengholt both point to DDD regarding how to organize data in domains.[3] [3] And they both address that rescoping DDD for data and not software is far from easy and ideal—although they provide sound and applicable advice on how to do it. Nevertheless, for a data catalog in particular, I suggest taking another approach in understanding and designing domains, one that focuses less on software and more on knowledge. This approach is found in information science.

I'll first explain domain thinking in information science, and I'll then briefly discuss the differences between domain thinking in DDD and information science in a note.

In information science, a domain has no links to a technological reality per se. It's purpose is not to create software; it simply focuses on people and what they do, and it defines a domain like this:

> A domain, then, can be a group of people who work together if they share knowledge, goals, methods of operation, and communication. It can be a community of hobbyists, a scholarly discipline, an academic department, and so on.[44]

This is the definition of domain that I rely on in this book, and behind this very simple definition, it follows that a domain is made of:

- *Knowledge* as an ontological base,[5] meaning a shared understanding of concepts and their relations
- *Goals* as a teleology,[6] meaning that this group shares ambitions of what they want to achieve or obtain—what drives them
- *Methods* of operation, meaning hypotheses and methodologies to test and expand the domain
- *Communication* as social semantics, in the sense of what tools and systems the group uses to communicate

3 Zhamak Dehghani, *Data Mesh: Delivering Data-Driven Value at Scale* (Sebastopol, CA: O'Reilly, 2021). Piethein Strengholt, Data Management at Scale (Sebastopol, CA: O'Reilly, 2023, second edition), p. 35-49

4 Richard P. Smiraglia, *The Elements of Knowledge Organization* (Cham: Springer, 2014), 86. The theoretical definition—cited from the same source—goes like this: "A domain is a group with an ontological base that reveals an underlying teleology, a set of common hypotheses, epistemological consensus on methodological approaches, and social semantics."

5 *Ontology* is the study of what exists (implicitly, in a data domain). See: *Ontologies*, *https://www.isko.org/cyclo/ontologies.htm*

6 *Teleology* is the study of what the intrinsic purpose inside something is (implicitly, in a data domain).

Each domain also has various degrees of *intension* and *extension*.[7]

The *intension* is how deep a domain goes in terms of the level of expert knowledge. For example, academics would have a deeper intension than hobbyists. Intension has a very concrete meaning: a domain can have an infinite layer of subdomains, so, for example, winemaking can have subdomains such as natural winemaking and traditional winemaking. In the case of the hobbyists, the intension stops there, but that would not be the case for the academics, who would further divide the intension of natural winemaking into organic winemaking and biodynamic winemaking and probably even further.

Extension, on the other hand, refers to the level of breadth in the domain, and in this case, the hobbyists have a broader extension as they are likely to include adjacent domains into their own domain, without mastering them at a professional level. Winemaking would be part of a domain also comprising travel and pleasure, for example.

The fundamental problem of applying DDD for data, in a data mesh, is that intension and extension are not free. Domains of various levels of intension and extension are put together in a string to define how data flows between software components. That may work for a data mesh orchestrated in an actual IT landscape, but it will not work in a data catalog: it will not deliver the total overview of data in your company in a structure that is searchable enough to create data discovery. To understand the difference, take a look at the domain mappings in Figure 2-3, and compare them with the data lineage depictions. If we were to accept only DDD for data as structuring our domains at the metadata level, then we would have to rely only on data lineage. This is technically correct but is conceptually confusing, as domains and subdomains are forced to structure themselves around the physical movement of data, and not the conceptual organization of data, thematically, in domains.

In Figure 2-3, I have added intension and extension, to our domain architecture diagram. These elements, which constitute a domain in information science, are all placed where they belong in the domain architecture of the data catalog. At this point, we are ready to run through the domain architecture in a data catalog, and I begin with the layer that can be structured based on either processes or capabilities.

7 Joseph T. Tennis, "Two Axes of Domains for Domain Analysis," *Knowledge Organization* 30, no. 3 (July 2003): 191–95.

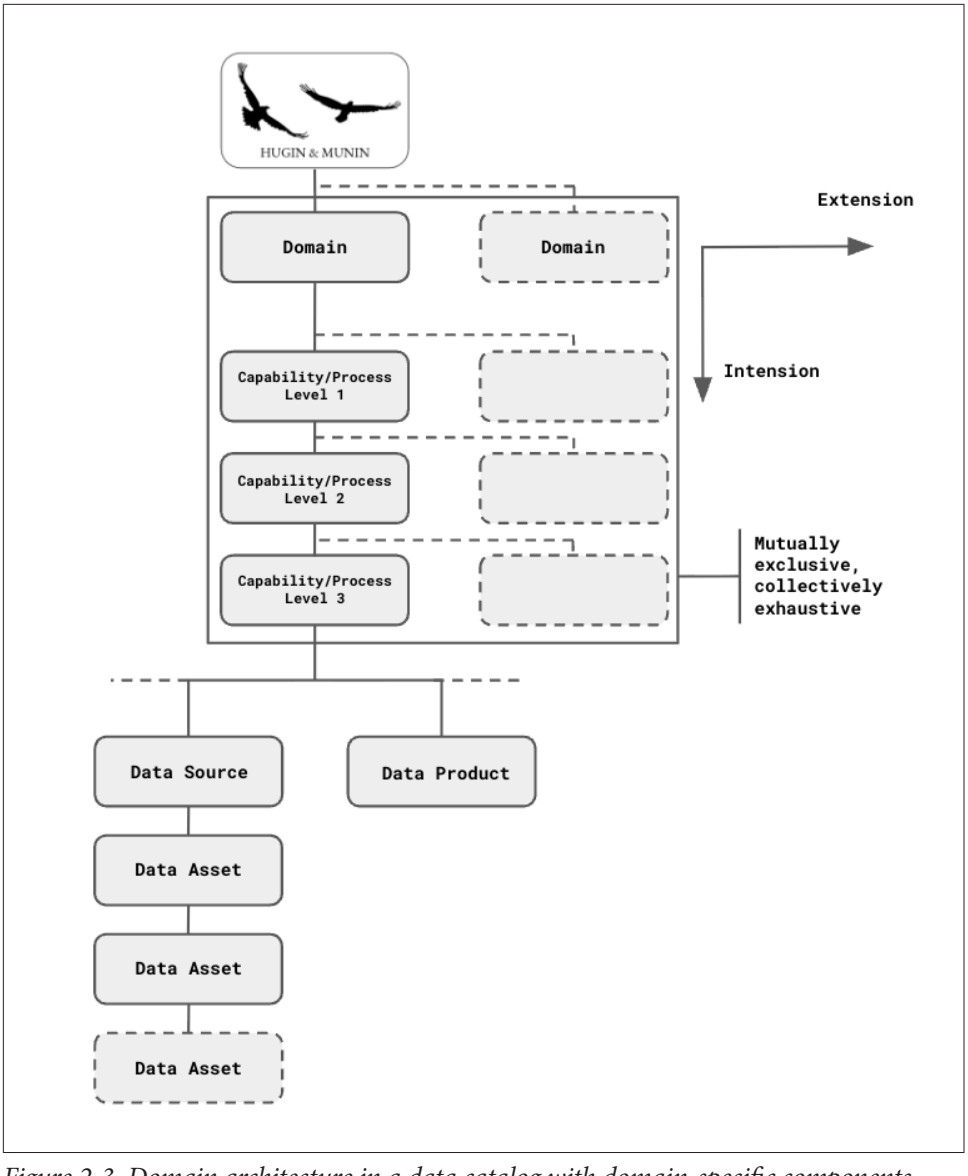


Figure 2-3. Domain architecture in a data catalog with domain-specific components

Processes and Capabilities

In this section, I am organizing data in a structure that represents the company in which the data is created. This structure is defined by either processes or capabilities.

Processes describe *how* a company performs its tasks. It's how things are done. Capabilities describe *what* tasks a company performs—what the company is capable of. Neither processes nor capabilities reflect the business units of your company 1:1.

Furthermore, processes are part of capabilities: capabilities consist of people, processes, and technology.

You can find a great introduction to processes and capabilities in *A Guide to the Business Architecture Body of Knowledge* (BIZBOK),[8] [8] which I recommend you read before you actually begin mapping domains in your data catalog.

I suggest you organize by capabilities if your data catalog is closely connected to the enterprise architecture activities, which rely on capability to manage the IT landscape. I suggest you use processes if your company already has a highly controlled process map.

The very first step to organizing your domain is to choose between creating the domains as processes or capabilities. Both will work fine, as they are stable entities. In either case, you start with high-level processes or capabilities and divide them into various sublevels.

A *process* domain is put together based on *how* things are done. Processes are part of a value chain that expresses *how* the products or services of a company are created. Processes are either directly or indirectly part of this value chain. Direct processes are, e.g., Research & Development, Manufacturing, and Sales. Indirect processes are supportive processes or strategic processes that enable the value chain, and they contain processes within themselves also.

In Figure 2-4 you can see an example of an indirect process map in the Hugin & Munin data catalog, namely HR processes. Pay attention to the process aspect in the level just below "HR Processes" that expresses how employees join, work in, and leave a company in the following process steps: Recruitment, Onboarding, Development, Self-Service, Offboarding, and Resignation. As you can see, all groupings are part of an overall process. Therefore, the domain depicted would partly lose its meaning if one of these parts were missing.

The knowledge, goals, and methods must be described in each part of the domain at the process level. Ideally, the processes displayed are self-explanatory, but you must ensure as logical an overview as possible for the end user of the data catalog.

8 See *A Guide to the Business Architecture Body of Knowledge* (Business Architecture Guild, version 11), specifically section 2.2 for capabilities and section 3.4 for processes.

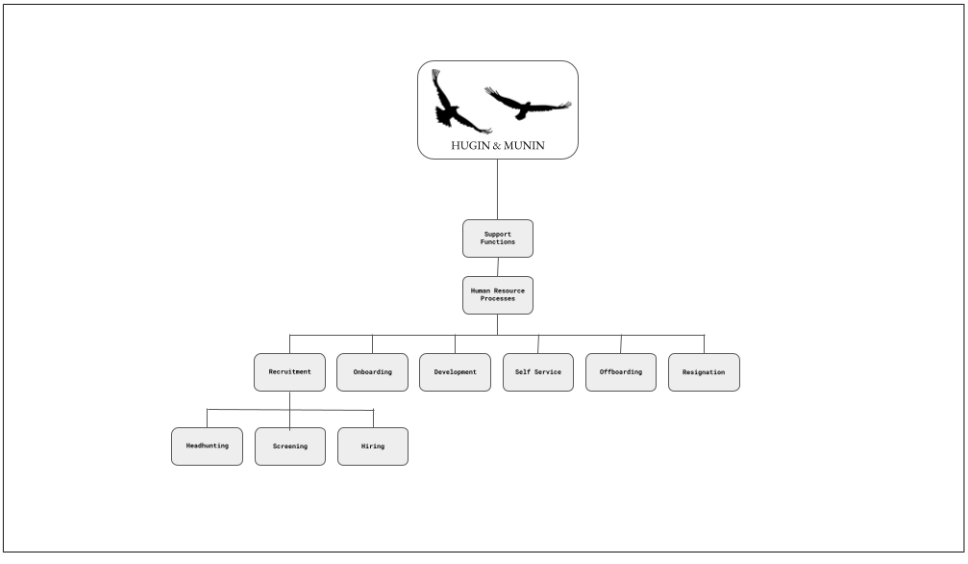


Figure 2-4. Domain architecture based on processes

A *capability* domain is put together based on *what* things are done. Capabilities are performed by many different business units and are therefore present in many different business processes. Unlike processes, capabilities are expressed using nouns, also for activities performed, typically coined as Management or Analytics, Determination, and Prioritization.

In Figure 2-5 you can see an example of a capability mapping in the data catalog, in this case, Data Analytics. I advise you to look closely at the level just below Data Analytics, containing the capabilities: Descriptive Analytics (what happened?), Diagnostic Analytics (why did it happen?), Predictive Analytics (what might happen?), and Prescriptive Analytics (what should we do because of what will happen?).[9] [9] Unlike the processes you saw in Figure 2-4, the capabilities are not part of a chain of events—they are not steps in an overall process. They can be performed by many different parts of your company, simultaneously and independently.

9 A discussion of these capabilities can be found at "4 Types of Data Analytics Every Analyst Should Know—Descriptive, Diagnostic, Predictive, Prescriptive".

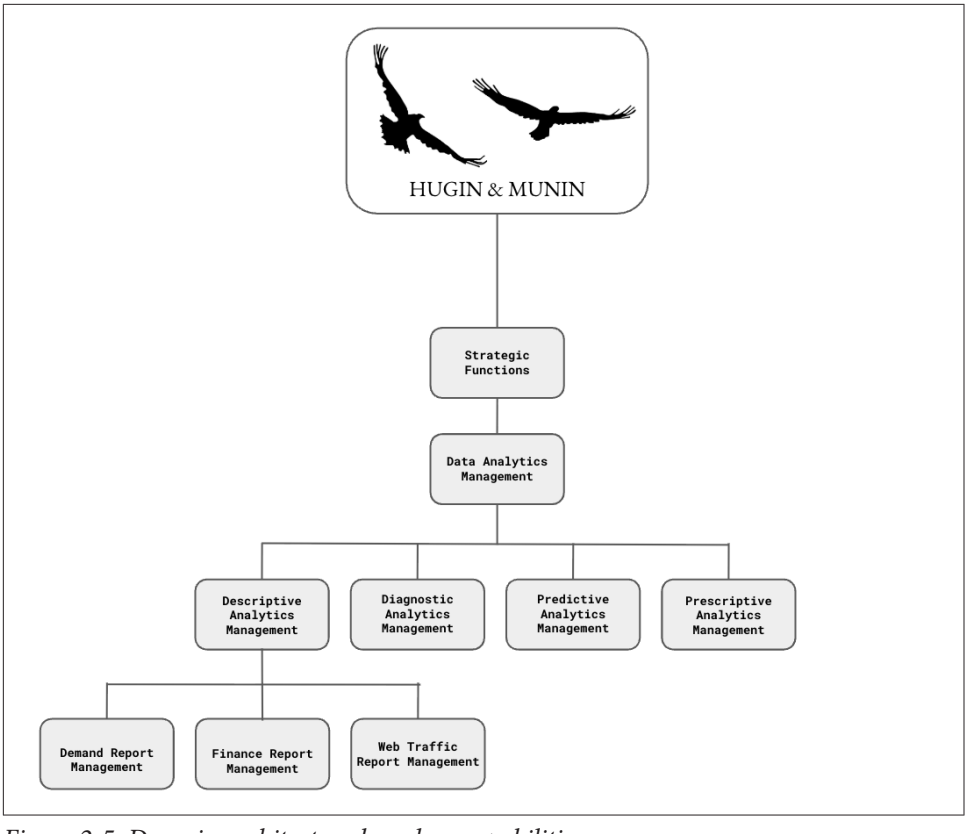


Figure 2-5. Domain architecture based on capabilities

Do not be tempted to build your domains directly based on your organization diagram. It's tempting to do so, because you have it at hand and mapped out for you already. But you must keep in mind that your domains should be stable entities to land data safely in the data catalog. And organizations change all the time: teams are merged, split up, outsourced, re-created, and reorganized constantly—and you end up maintaining a continuously changing domain architecture instead of catering to data discovery.

If you work in a highly regulated industry, such as food, pharma, or oil and gas, your company has an official process map with several layers, contained in a quality management system (QMS).[10] It's a requirement for regulated industries to document their processes, as these industries must explain and show proof of correctly performed processes under audit and inspections from the authorities, such as the Food and Drug Administration (FDA) in the United States. If your company has such a process map, use it! You must organize your domains in the data catalog so that they mirror the process map 1:1, but you can leave out the lowest levels of the process map. They are typically too detailed and explain very specific actions by employees. Below the process levels, you should define data sources—take a look at Figure 2-4 if this puzzles you.

If your company does not have an existing process map, you can create either a process map or a capability map. You must remember not to mix processes and capabilities, as they are different in nature. You must choose one or the other and stick to that.

Now we move to the level below processes/capabilities, which is the technology layer —your actual data sources.

Data Sources

When you have successfully mapped your domain into processes or capabilities, you move deeper into the architecture and depict the data sources that support them. The generic data source is a technology component. These technologies can be databases, data lakes or data warehouses, and actual applications.

Keep in mind that you cannot skip mapping domains in either processes or capabilities. Your map of data sources will be meaningless, close to unreadable for end users, if you move directly from the data catalog main entry and into generic data sources. Even the data discovery team will lose track of what data sources are registered in the data catalog if you proceed in such a way.

Let's continue with the Hugin & Munin capability domains example from the previous section. Take a look at Figure 2-6. Hugin & Munin uses Power BI as the data source to support the capability of *Demand Trends Report Management* (part of *Report Management*, part of *Descriptive Analytics*, part of *Data Analytics*). This is first registered as a generic data source, as there are many specific instances of Power BI,

10 QMSs are specified in standards and must adhere to, e.g., ISO 9001:2015.

so you must remember to divide your data sources into generic and specific ones. The generic data source simply refers to the software component, such as Tableau, Qlik Sense, or, in this case, Power BI.

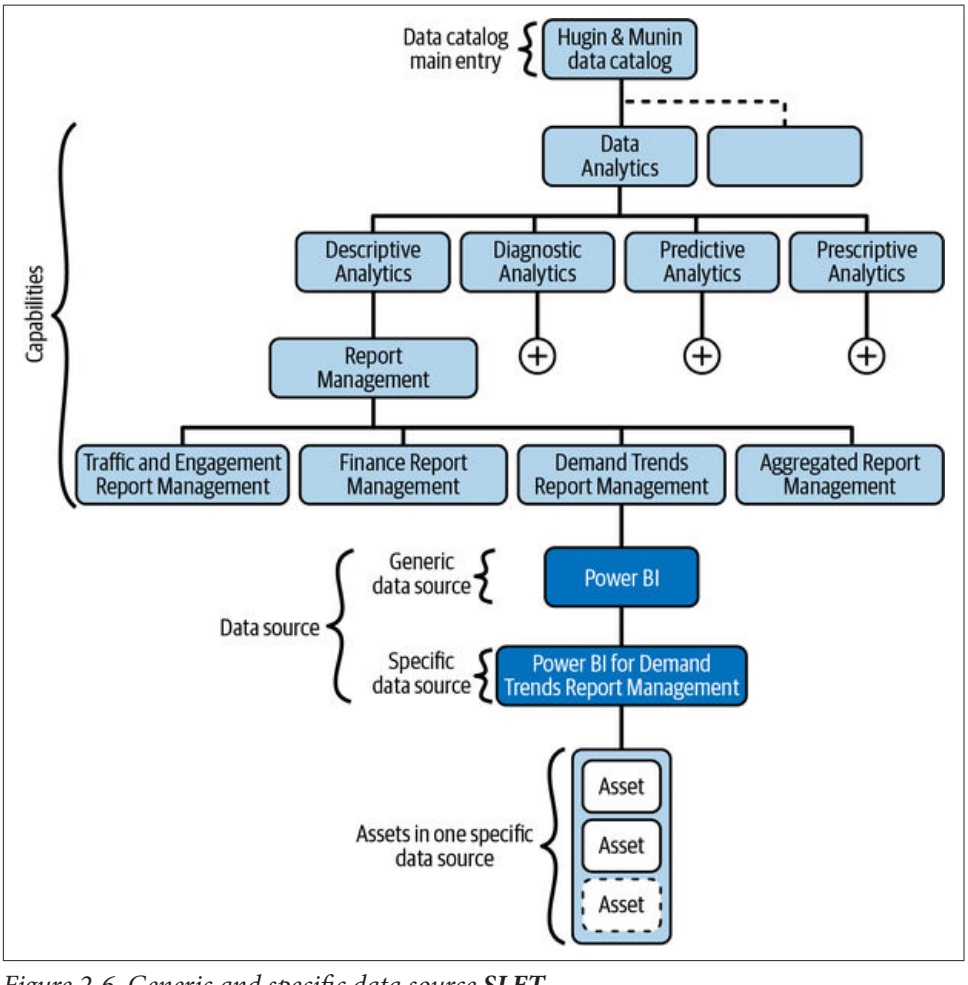


*Figure 2-6. Generic and specific data source **SLET***

You should not ignore creating the layers above the data sources, depicted in Figure 2-6. That comes with the risk of losing the capability to search for data, as the lack of organization makes it hard to understand where in the company the data comes from.

Below the generic data source is the specific data source. You need to treat each data source as a specific instance of a generic data source, so that the data contained in the

capability are the only ones supporting this capability. This will also ease how you assign roles and responsibilities to the data.

A specific data source simply means that it is a specific instance of the generic data source. In this case, it's a specific subscription of Power BI. You must be aware that it could also be only part of a subscription; sometimes it's not relevant to expose all data.

Think of this from a domain perspective: your data sources are how the domain communicates. So how many data sources you have, and how much data from them you include in the domain, must be no more, no less, than the number of sources the domain uses to communicate.

The possibility to divide data sources like this is another advantage of an information scientific approach to domains. DDD domains always struggle with how to handle data sources (software) used in several domains, because they are rooted in software process thinking.

At this point, you have created a safe landing zone for data sources. Figure 2-2 to Figure 2-6 illustrated how you organize data: first in domains based on capabilities or processes, then in generic data sources, and finally in specific data sources, where you will place your collection of data.

The first thing you need to do is to pull or push your data from the data source into the data catalog.

Organizing Data in the Domains

How well you organize your data inside their domain will determine if your data catalog is a success or a failure. Data that are not assigned relevant metadata will slowly disappear in the total amount of data—they will not appear in searches by the end users of the data catalog. You need your data to be discovered and used, and this section will teach you how.

Metadata for Data

I defined *data* in Chapter 1 as an entity of data that exists in your IT landscape. It could be a file, folder, or table, stored in a data source such as an application or database, etc. The data in the data catalog consists of metadata that represents that file, folder, schema, and so on in the data source.

Metadata is popularly defined as "*data about data.*" This definition lacks the core characteristics of metadata, and so I therefore define metadata as *Data that exists in*

two places at the same time (at source and in a tool pointing to those sources, i.e. the data catalog).

To elaborate, metadata *refers* to other data, and it does not exist without the data it refers to. Data in the data catalog is made up entirely of metadata; all your data in your data catalog *refer* to data in data sources. In the sections that follow, you will learn to describe your data so that you maximize their data discovery potential.

All data have owners and stewards. Consider these roles as mandatory metadata: all data need to have assigned owners and stewards. *data owners* are the actual owners of data in the source system. It will always be the data owner who defines who can access a data source and what data can be used for.

It's a strategic gain for your company that the data catalog can help assign data ownership. This is a very difficult task in most companies, due to the common lack of understanding of this responsibility—it's complex and only offers hardship. With a data catalog, that's different: data ownership suddenly comes with services such as an overview of sensitive data and a control mechanism of *how* data owners share data.

In daily operations, it will be the *data steward* who maintains data in a domain. Specifically, the data steward adds metadata and handles data access requests and lifecycle management activities in general—check out Chapter 7 about lifecycles.

To properly organize the data in a domain, you need to think about how the metadata for each data was derived or added to the data catalog. The metadata for data can be derived from a data source, it can be added when the data is already in the data catalog, or both.

On a more general level, data can have:

- Metadata derived from the data source
- Metadata added in the data catalog
- Metadata *either* derived from the data source *or* added in the data catalog

Let's go through each scenario.

Metadata derived from the data source

There are two types of metadata in an data that are always derived from the data source:

- Technical metadata
- Business metadata

Technical metadata tells you exactly what data source the data is stored in, who created the data, when the data was created, the file format of the data, etc. Its metadata

that is automatically attached data, during its creation and existence. For example, a Qlik Sense report created by a business analyst in the HR department on December 13, 2021, based on JavaScript and QEXT files.

Business metadata is metadata that describes the data in human language, for example names of tables and columns, descriptions and definitions of data types, etc.

Derived metadata constitutes the minimum description of your data. It's valuable, and you need it to get the facts right about your data, but you can't rely on derived metadata alone: it will not contextualize your data sufficiently to make your data catalog perform relevant search experiences for end users—people won't find what they are looking for. Therefore, you need to add metadata to your data inside the data catalog.

Metadata added in the data catalog

You can add metadata such as descriptions, people, and glossary terms to the data in your data catalog. You can also modify the metamodel of the data catalog to better make it express the logic of your company (I discuss classifications separately, later).

Descriptions are high-level descriptions of data, and they should contain at least two elements: primary and secondary usage. *Primary usage* is a brief explanation about what the data is used for in the data source where it was pulled/pushed from. *Secondary usage* is suggestions from the data provider to potential consumers about what the data can be used for.

People are all the relevant persons who should be listed in the data catalog. These are, for example:

- Domain owner
- Domain steward
- Data source owner
- data owner
- data steward
- Term owner
- Term steward

The *domain owner* is the owner of a given domain and is responsible for managing the domain. Typically, this person will also be who data management usually thinks of as a *data owner*, as this role spans several data sources. The *domain steward* is responsible for curating a domain and providing access to sources. The *data source owner* is the owner of a given IT system. The *data owner* is the person who has created the data. The *data steward* is typically responsible for managing the practical work of managing many data. The *term owner* owns a domain term or a global term.

Free glossary terms do not have ownership, as they are unmanaged. The *term steward* manages several terms.

Glossary terms are basically words that are found in various glossaries inside the data catalog. Glossaries allow you to "tag" your data with terms. This increases the discoverability of the data when users search for topics where the data could be relevant. The glossaries are lists of words that describe your company, and the glossaries are controlled to various degrees, by either a domain glossary team or a centralized global glossary team.

Figure 2-11 illustrates three different glossary types:

- Free glossary
- Domain glossary
- Global glossary

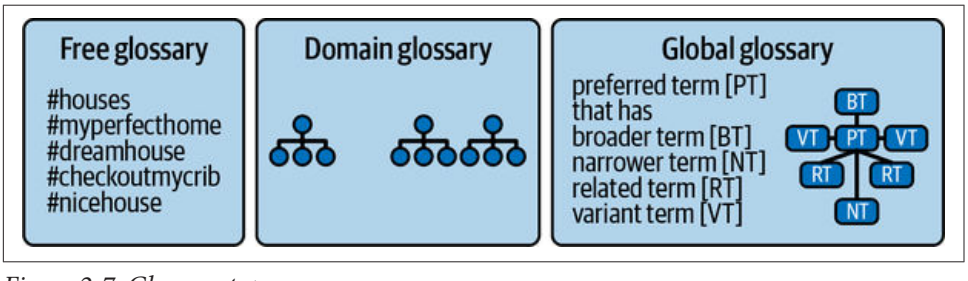


Figure 2-7. Glossary types

The *free glossary* is a folksonomy. Folksonomies are user-generated glossaries that organize data by the use of tags. If you use social media, you've probably seen posts relating to a certain topic marked with a hashtag. For example, #puppies for posts involving dogs, #ootd for posts about someone's outfit of the day, or more seriously, #metoo for posts about sexual harassment. There are no formal rules in a folksonomy; everyone can create tags and use them as they want. However, don't let its use on social media fool you into thinking they're just for fun. In a data catalog, you can use a folksonomy to describe data in completely unique ways, and this will enable you to search for specific topics that a central team would not have thought of.

A *domain glossary* is a taxonomy. Taxonomies have a hierarchy. They can be narrow/broad or have facets of narrow/broad relationships of terms. It is a user-generated glossary that organizes data by the use of terms that are agreed upon and aligned on the domain level. Taxonomies are more controlled than folksonomies. Taxonomies are created by a small group of people within a domain and not the general public—or the entire group of employees in a company—and, subsequently, the taxonomy creates a more formal language. Tags are completely free associations; for example, *#ootd* in the folksonomy might be *daily clothes* in a taxonomy. This is because a group

within a domain has decided a logical term to describe this specific element of clothing. Also, taxonomies tend to create hierarchies between terms, some being broader than others; for example, *daily clothes* is a narrower term than *clothes* but broader than *daily clothes when it rains*. Some taxonomies, especially in online shopping experiences, apply a faceted approach that enables the user to "build" a unique expression when organizing data and searching. Such facets could be colors: e.g., *blue* + garment, e.g., *dress*. This is certainly also applicable in data catalogs, and as you will see in Chapter 3, a faceted approach can be of great benefit. Taxonomies are domain specific. They represent a domain's formal description of itself in a glossary. The taxonomy alternative to the folksonomy is also very useful when searching the data catalog, as you will see in Chapters 3 and 4.

The *global glossary* is a *thesaurus*. It's a structure that moves away from hierarchy thinking and toward cluster thinking instead. It has a center, the preferred term (PT). This is the core of the cluster, the main thing that is described. Let's stick to the above example and say that the preferred term is *daily clothes*. This preferred term is surrounded by variant terms (VTs), which are synonyms. In this case, a VT could be *#ootd*. There are also more freely associated terms, called related terms (RTs). In this case, it could be, for example, *raincoat.* Finally, there are narrower terms (NTs), for example, *daily clothes when it rains*, and broader terms (BTs), in this case *clothes*. But keep in mind that it is the PT that is the center of the cluster; it may have as many BTs as you see fit, and it does not belong in an overall hierarchy. An ontology is highly controlled and has a lot of potential for improving search, as you'll see in Chapter 3.

From a systemic point of view, you lose the ability to analyze and improve search behavior if you believe you have perfectly described the truth of your company in a highly controlled glossary. You must strive for both no control and control at the same time.

Do not assume that more-controlled glossaries are less biased than glossaries where you have little or no control over the terms. What you get with control is a more consistent semantic expression, globally in your glossary, between your terms. But it will be just as subjective and biased a glossary as a loosely controlled glossary. For example, Melissa Adler has succinctly examined the gender and race bias in the cataloging practice in the Library of Congress in the United States.[11]

Do not take the implementation and management of your glossaries lightly. They are the cornerstone in improving your data catalog's search capability. You'll learn the organizational details in the rest of this chapter, and you'll realize how in Chapter 3.

11 Melissa Adler, *Cruising the Library: Perversities in the Organization of Knowledge* (New York: Fordham University Press, 2017).

A glossary is not to be confused with a *data dictionary*. The data dictionary is a basic tool that specifies the types of data you have at a generic level, typically by listing the field name and providing a description of the kind of data the field contains. The data dictionary is a natural part of data catalogs and may not always be a separate feature, but simply included directly in the data. A glossary, on the other hand, is an interpretation and reflection of the data that contextualizes this data into the overall knowledge of your company.

Knowledge Graph Powered Data Catalogs

Finally, you can change the *metamodel*. In Chapter 1, I mentioned that data catalogs have flexible metamodels. Metamodel flexibility can be provided at various levels. The most advanced and useful level of metamodel flexibility is provided in knowledge graph–based data catalogs - and these are indeed becoming dominant in the industry, because of their proven superior functionality. Basically, your data catalog must be built on a graph database, to be knowledge graph native. The metamodels of knowledge graph powered data catalogs have these characteristics:[12]

The metamodel is visual.
: It's a browsable metamodel that you can see—a structure that shows how all entities are connected. (certain metamodels need to be queried in a graph language to be visualized, other knowledge graph powered data catalogs have that as a default - and useful! - feature)

The metamodel is extendable.
: You can add new entities to your metamodel and likewise create new relations.

The metamodel is searchable.
: This means, basically, that you can query everything in the data catalog using a database query language like SPARQL, Cypher, GraphQL etc.

In the context of this section, you should consider that the metamodel is extendable. This means that you can add entities to the metamodel. For example, in the case of Hugin & Munin, the metamodel could be extended with entities that represent data from the people and systems that treat wood, from the moment it is cut, to how it is inventoried as lumber, to the final building in which it is used.

12 Juan Sequeda, "What Does It Mean for a Data Catalog to Be Powered by a Knowledge Graph?", September 2022, Datanami.com.

Adding a specific business context is useful for the metamodel, because you can now link all the data that is associated to this business context. This will improve your ability to organize data and search for it.

Data lineage and semantic relations as graphs are important metadata to each data. I defined both of them in Chapter 1, but let me briefly recap. *Data lineage* shows how an entire data travels horizontally, in an ETL/ELT (extract, transform, load/extract, load, transform) process. *Semantic relations* show parts of data asset (e.g., a specific column in a table) and how it relates to parts of other data.

Both lineage and semantic relations are technical features. At a bare minimum, lineage visualizes ETL jobs in certain data sources, and semantic relations are built on a graph database that can be combined with natural language processing and machine learning in cases where it is derived from the data source. You should be able to expand both, manually and via API, inside the data catalog.

Knowledge graphs are key elements for AI in our era. I discuss this in chapter 11. take a look now, if you are curious!

Data Assets and Data Products

Figure 2-12 shows the data asset architecture of Hugin & Munin. Notice that it builds on Figures 1-3 and 2-3, which showed a fully organized data and a domain architecture built on capabilities. What you see is generic Power BI data, including all the metadata types I have discussed in this section. The straight lines from the central data and outward are lineage, and they depict the lineage of the entire dataset, where it comes from and where it travels further. The curved lines from the central data and outward are graph relations, and they depict parts of the data being conceptually/semantically related to other data.

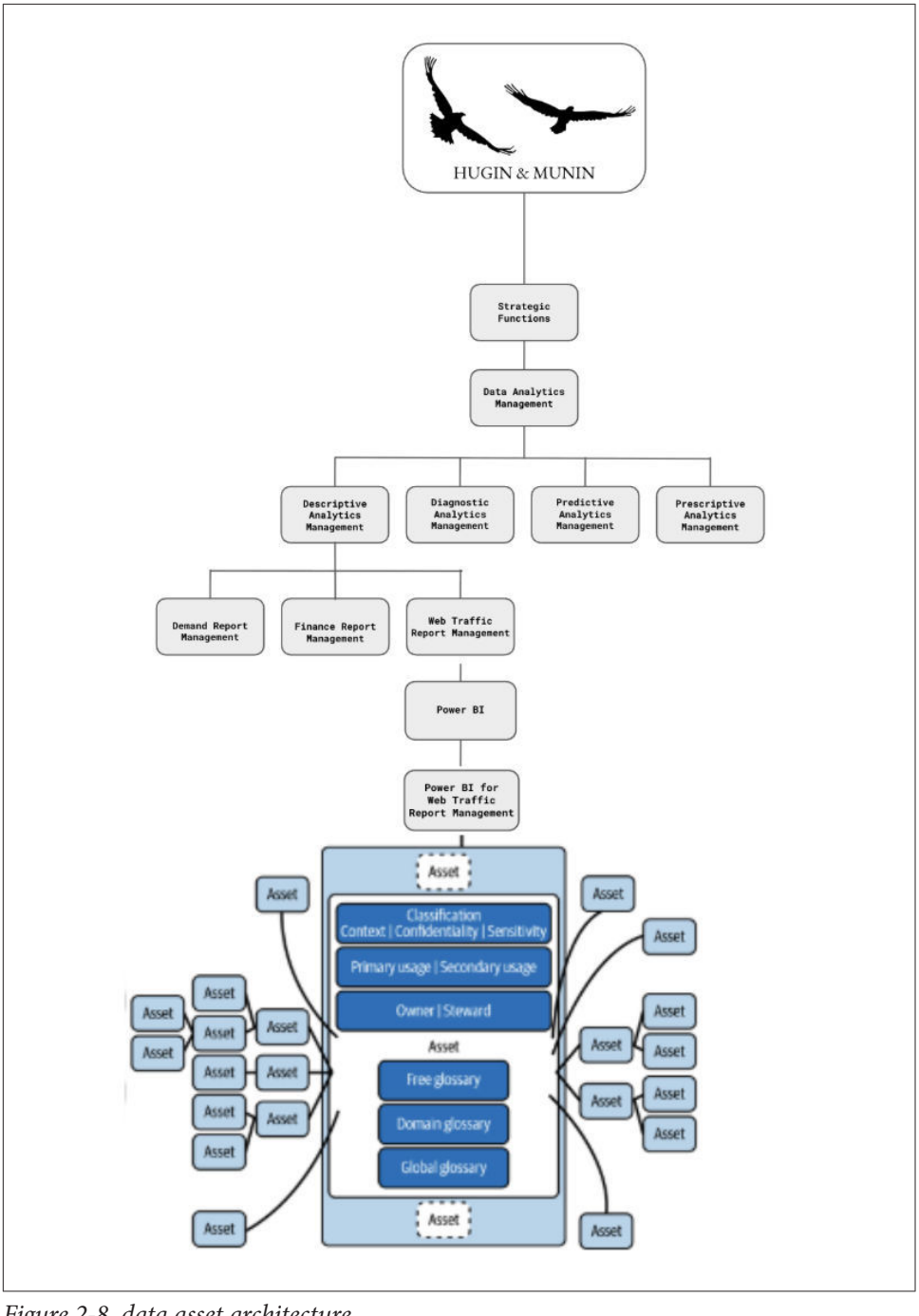


Figure 2-8. data asset architecture

Figure 2-10 shows a data product in a domain in Hugin & Munin. We will go through data products in chapter 8 - there is a lot to cover, in fact, data products are revolutionizing data engineering! But at this point, notice already that data products belong directly in a domain. Unlike data assets, data products are detached from any given application, in this case Power BI that you saw in Figure 2-8 above. Data products are completely autonomous containers of data - they exist to reduce the time it takes to find, access and use data in a company, which is why they are being built and implemented in companies world wide.

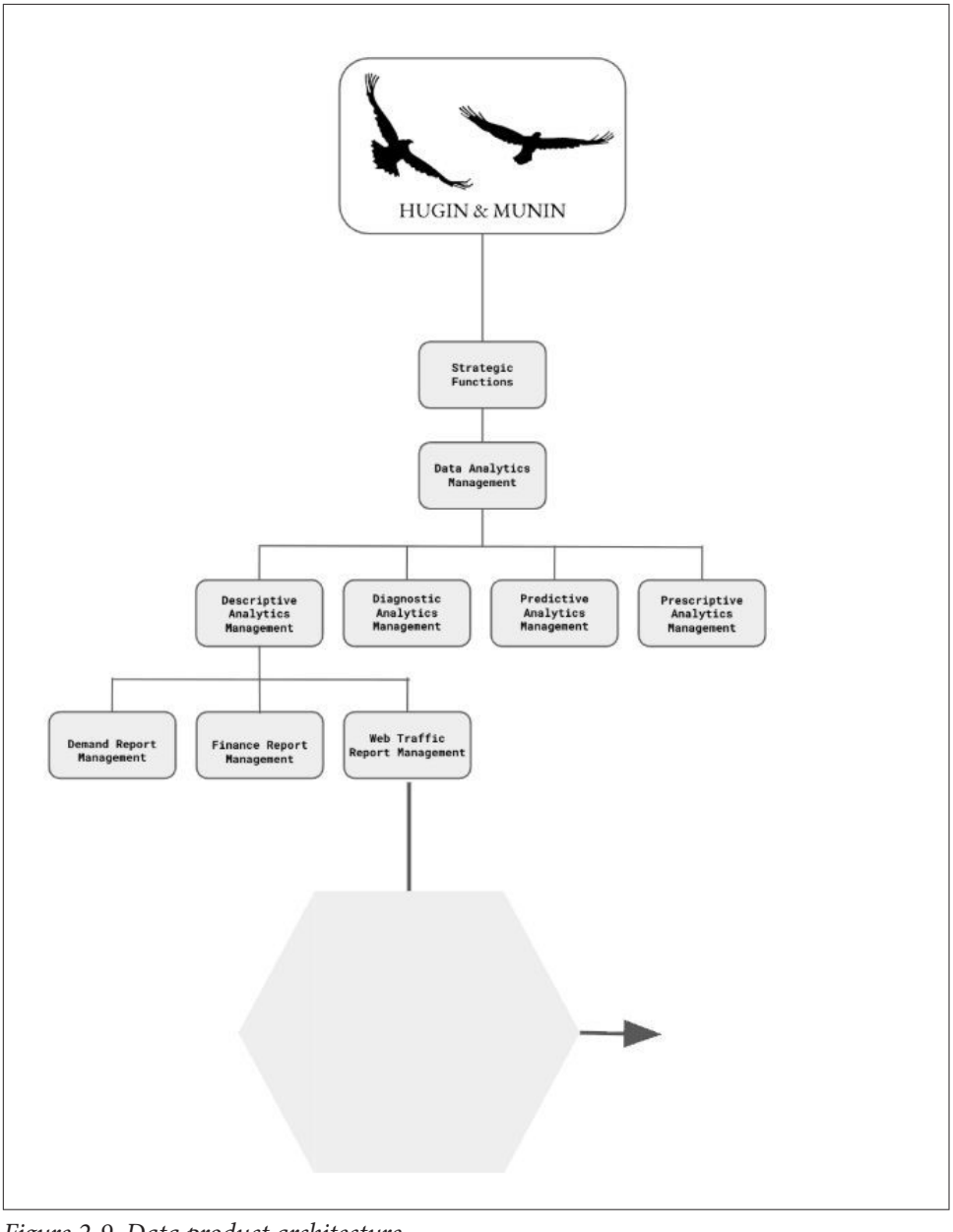


Figure 2-9. Data product architecture

Metadata Quality

In this section, I discuss how to analyze the quality of domains and data in the data catalog. Because everything in the data catalog consists of metadata, I call this analysis

metadata quality.[13][14] Metadata quality refers to how well domains and their data are represented. Without metadata of descriptions, people, and glossary terms, the search functions in your data catalog will not work well.

Your ability to search a data catalog depends on the quality of this metadata—not on the quality of the data in your data sources!

Earlier in this chapter, I discussed intension (depth) and extension (width) of domains. These two things constitute the *actual* depth and width of the domain—how the domain happens to be architected in your data catalog.

But there are also *ideal* depths and widths of domains. They are not universal (no universal, ideal domain architecture exists). The ideal depth and width of a domain is based on a domain analysis and is expressed with one term, for both depth and width: exhaustivity.

Domain analysis is an essential part of information science. If you need to perform a domain analysis, take an initial look at Smiraglia's book, which I quoted earlier in this chapter. It contains a list of the methods usually applied.

Exhaustivity characterizes a state where a domain is perfectly expressed in depth and width. Exhaustivity, therefore, can be high if the domain is depicted in perfect depth and width. Likewise, exhaustivity is low if the domain is depicted in insufficient depth and width. This goes for its structure in the overall domain map, and it goes for all its associated metadata.

Let's say, for example, that the only global glossary term for the building material in Hugin & Munin is Wood. This is an unacceptable, low level of exhaustivity. All the buildings that Hugin & Munin design are made of wood, so the glossary would need to go into depth about what kind of wood was used for which buildings. Was it beech or pine for the family house just outside Oslo? Was it oak for the skyscraper in downtown Stockholm? The exhaustivity of the glossary needs to align with the domain in scope. If the exhaustivity is too low, then the glossary can't be used, functionally, to depict the domain, and then searching for data won't work. But even if the level of

13 I do not discuss how you analyze the data quality of the data sources that the data catalog depicts—it's another discipline that you can find numerous sources on. In most of those standards, data quality and metadata quality are interwoven and made measurable, e.g., FAIR (findable, accessible, interoperable, and reusable).

exhaustivity is perfect, things can go wrong. You need to get familiarized with specificity also.

Specificity is the *usage* of the *actual* intension (depth) and extension (width) in the domain—and not a potential usage of the exhaustivity. So if you, e.g., have only pulled or pushed data sources to a few of the actual subdomains in your domain, then exhaustivity is high and the specificity is low. Likewise, if you have pulled or pushed data sources to most subdomains in your domain, then your specificity is high.

Let's return to the example above: Wood. Say that the global glossary contains the words Beech, Oak, Pine, and Wood—and that this level of exhaustivity is acceptably high for Hugin & Munin. Now, what happens if the only glossary term that data have been tagged with is Wood? Then, exhaustivity is high, but specificity is low. This means, basically, that your search features could have worked because the data could have been tagged with the most appropriate terms from the glossary (the exhaustivity is high), but that search does not work because those glossary terms were not applied.

If the glossary terms for a specific domain cover the domain appropriately, then the level of exhaustivity is high. If they do not, then the level of exhaustivity is low. This is the case if the glossary terms are too broad or simply lacking. If all or most of the glossary terms for a specific domain are applied to some of the data in the domain, then the specificity is high. If they are not, specificity is low.

Figure 2-13 provides an overview of how to think about exhaustivity and specificity. It's a matrix, because you basically have four scenarios for exhaustivity/specificity, either both are low or high, or one is high and one is low. We'll return to this specificity/exhaustivity matrix in Chapters 3 and 4.

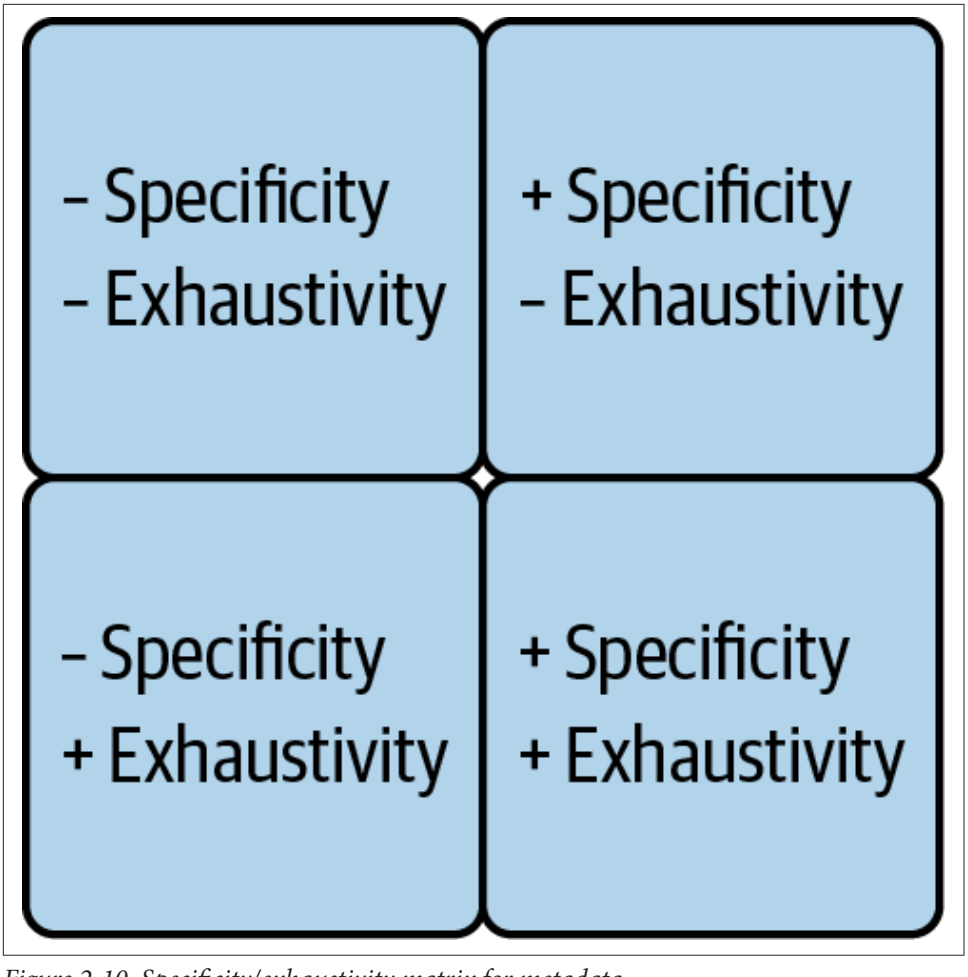


Figure 2-10. Specificity/exhaustivity matrix for metadata

Exhaustivity and specificity are important for search. We'll talk about it more in Chapter 3, but here are some questions for you to think about in the meantime: Is it easy to search for a very special type of data, if it has been cataloged with only general, broad glossary terms although more specific terms exist in the glossary? Is it easy to browse your way to a very special type of data in a domain, if the domain only has few, very broad subdomains? Can you trust that you have searched correctly, if the only hits you get on something very general are detailed, specialized data?

Classification

The metadata of a data catalog may include metadata from the data sources themselves, as well as glossary terms added in the data catalog, but it also includes the classification of data. So, what does it mean to classify something?

Classification can mean several things, depending on whom you ask. For example, you might hear the following dialogue between a CISO and a DPO:

> CISO: "I have classified this data … and it is highly confidential!"
>
> DPO: "What do you mean? I classified this data months ago; it's not sensitive at all."
>
> CISO: "What? I completely disagree!"
>
> DPO: "Huh? So do I!"

In fact, both the CISO and the DPO have correctly classified the data—it's often the case that data is classified as highly confidential and not sensitive at all, at the same time. This is exactly because classification can mean different things. There are three types of classification:

- Content
- Confidentiality
- Sensitivity

In your data catalog, you must enable all three types of classifications on each data, and users must be able to combine them as they want. Remember: high levels of confidentiality do not imply high levels of sensitivity, and vice versa.

Classification of content refers to what your data is. It's the unique label of your data that defines exactly what it is about. I advise you to build the content classification in the data catalog based on the structure of your domains—and that you formalize this in instructions to domain owners, so that the content classification is consistently applied. For example, the following expresses the classification of content for an data in the Hugin & Munin data catalog shown in Figure 2-6:

DA.DeA.RM.DT.Power BI

The logic of this expression is that all the capabilities as acronyms are shortened so they are all distinguishable from each other, combined with the generic data source. You can see the expression visually explained in Figure 2-14.

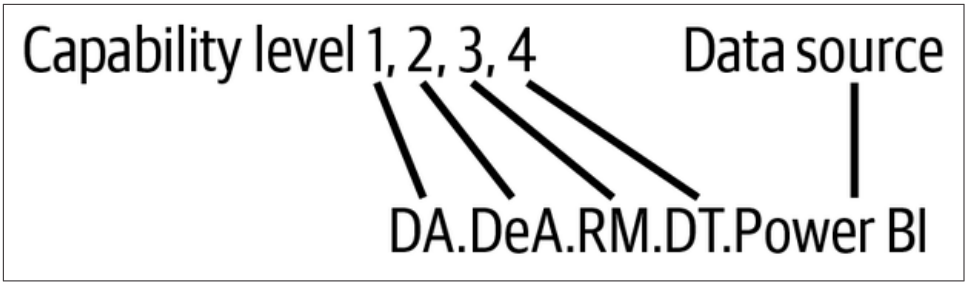


Figure 2-11. Methodology for content classification

So, this data is within the Data Analytics (DA) domain, under Descriptive Analytics (DeA), under Report Management (RM), within Demand Trends (DT), and uses Power BI as the data source. As you will learn in Chapter 3, content classification is a very powerful element to search your data catalog. You may find it cumbersome to define content classification for each specific subdomain of data and make users apply it, but these processes can be automated in data catalogs. And the payoff really is remarkable.

Classification of confidentiality is how secret your data is. All your data have a level of confidentiality. Your CISO is the defining authority of confidentiality, in close collaboration with the legal counsel. Confidentiality classifications originate from military, police, and intelligence organizations. The North Atlantic Treaty Organization (NATO) has the following confidentiality classifications:

- COSMIC TOP SECRET
- NATO SECRET
- NATO CONFIDENTIAL
- NATO RESTRICTED

Also, the label UNCLASSIFIED is added to open, public information in NATO. The level of confidentiality is determined by the level of damage a breach of the data would cause. The exposure of COSMIC TOP SECRET data from inside NATO will cause *exceptionally grave damage*, NATO SECRET will cause *serious damage*, NATO CONFIDENTIAL will cause *damage to the interests of NATO*, and finally NATO RESTRICTED will *be disadvantageous to NATO*.

Although it's generally good practice to err on the side of caution, the extreme can be detrimental.[14] If all data in your data catalog is TOP SECRET, then you end up with a

14 The most tragic example of that is the terror attacks in the United States on September 11, 2001. The Congressional Research Service concluded in 2011 that the terror attacks could likely have been avoided if a less restrictive confidentiality classification had been applied in various ways between the CIA and the FBI.

cumbersome security structure, and you lose the point of the data catalog: to make data discoverable, and therefore used in new contexts. You need to be realistic about how confidential your data actually is. Think about it this way: if the data were to be accidentally released, how much damage would it cause? A document containing an SOP for cleaning a production facility is relatively harmless. A BI report containing the production performance of the production facility would be less harmless, but not life-threatening to the company. A drawing in DWG format of the production facility in complete detail would be catastrophic in terms of industrial espionage, if it were accidentally made public.

Classification of sensitivity is how personal your data is.[15] All your data have a level of sensitivity. It is the DPO who ensures that the various regulations on personal identifiable information (PII) are enforced in your company—as such, it is considered a core part of data governance. If your company is operating on a global scale, these are the levels of sensitivity you must choose between and assign to your data in the data catalog:

- Nonpersonal data
- PII
  - — Personal data
  - — Sensitive personal data

PII is a global term that merely distinguishes between data that is not personal and data that is. Personal data simply means that the data in question can identify a person. In the General Data Protection Regulation (GDPR), PII is further distinguished between personal data and sensitive personal data. Personal data is private, but not compromising; for example, name, address, and age. Personal sensitive data, on the other hand, is compromising data, and includes, for example, membership of a union, political party, ethnicity, and health data. Many data catalogs can automatically detect if a data holds a value that is PII, so this classification can be automated.

Your data catalog has a big selling point. Normally, your CISO and DPO will oppose solutions that let all employees see all data in the company. But as data catalogs contain only metadata, not only will the CISO and DPO be OK with this solution, they will support it, because they get something they didn't have before. The CISO can control confidentiality, and the DPO can control sensitivity on the actual IT landscape and not just in policies.

15 The reality is that the personal information can be difficult to ascertain, as several nonpersonal data together constitute personal information. See Sille Obelitz Søe et al., "What Is the 'Personal' in 'Personal Information'?" *Ethics and Information Technology* 23 (2021): 625–33.

Summary

In this chapter we discussed how to organize data in a data catalog. Here are the key takeaways of the chapter:

- Data Catalogs are being augmented by AI - you can significantly increase implementation success of data catalogs with AI features.
- Data sources in your data catalog are organized into domains. The information scientific understanding of domains is the one used in this book, as it is less software-centric and more conceptual than the one in domain-driven design.
- Above a generic value chain, the domain structure has three layers that consist of processes or capabilities, data sources, and data.
- Each layer has metadata.
- At the data layer, there are three types of glossaries, namely folksonomies, taxonomies, and ontologies, and they all play an important role in describing your data.
- Classification means several things, namely classification of content, confidentiality, and sensitivity. All three are combined freely, because all kinds of data can be, e.g., highly confidential and not sensitive at all.
- The data catalog format is a structure that in itself summarizes the content of this chapter, as it structures all metadata that is necessary to assign to an data in a data catalog.

In the next chapter, we will talk about how you search for data in a data catalog.

CHAPTER 3

Search For Data: Concepts, Features, Mechanics

A Note for Early Release Readers

With Early Release ebooks, you get books in their earliest form—the author's raw and unedited content as they write—so you can take advantage of these technologies long before the official release of these titles.

This will be the 3rd chapter of the final book.

If you'd like to be actively involved in reviewing and commenting on this draft, please reach out to the editor at *shunter@oreilly.com*.

You now know how to organize data in your data catalog. Now, you're ready to search your data catalog. But why should you search a data catalog? What is it, exactly, you are searching for in a data catalog? And how do you actually search? How powerful is your search? How can you browse through data? And what role does AI play - how is it transforming search? These are the questions I discuss in this chapter.

Here is some good news: AI has completely redefined how you search data catalogs, and very positively so. AI makes searching for complicated topics very easy, and even finding your way to the data catalog is getting easier. And finally, AI also makes it possible to search your data catalog in connection with other precious data sources.

Here is some less good news: In this chapter you will have to pay attention to some hard stuff - academic vocabulary, technical details, and complex mathematics. But! There is a payoff: This chapter will empower you to master all aspects of the ultimate feature behind implementing a data catalog: Search. So, sit tight and expect to get the secret behind strategic influence in your enterprise.

Accordingly, this chapter is divided into three sections:

- Concepts
- Features
- Mechanics

Let's begin!

Concepts

In this section we cover:

- searching in vs searching for data
- information needs
- serendipity (and zemblanity)
- promptism

Searching in Data Versus Searching for Data

What do you search for in a data catalog? The one key thing that you need to understand when working with a data catalog is that, when you search a data catalog, you are searching *for* data, not *in* data. There's a clear distinction between the two that will drive how you interact with the data catalog.

So, what's the difference between searching in data versus for data?

Searching *in* data is when we search in the actual data for something we want to know. For example, we might ask: "How many people looked at our website last Saturday night?" The answer to this is a value from access records: 1340 people looked at our website last Saturday night.

Searching *for* data is when we search for the sources that contain the data we need. For example, we might ask: "Where can we find data about traffic on our website?" The answer will be a data location. We are not searching *in* the actual data, we are searching *for* the sources that hold the data.

As you can see, searching in data and searching for data go hand in hand. Once we find the data we are searching *for*, we can search *in* that data.

Searching *in* data is done with a *database query language* (DQL), illustrated in Figure 3-1. A DQL enables you to write statements and query different database technologies in the database's database management system (DBMS). The possibilities are endless; you can combine as many types of data in as complex ways as you want, since the DQLs are capable of performing intricate mathematical query statements on

the data you're searching in. Quite logically, data science has emerged from this. Data scientists search directly in data, using DQLs at an advanced level. Data science looks for patterns, connections, and correlations in (very) large amounts of data.

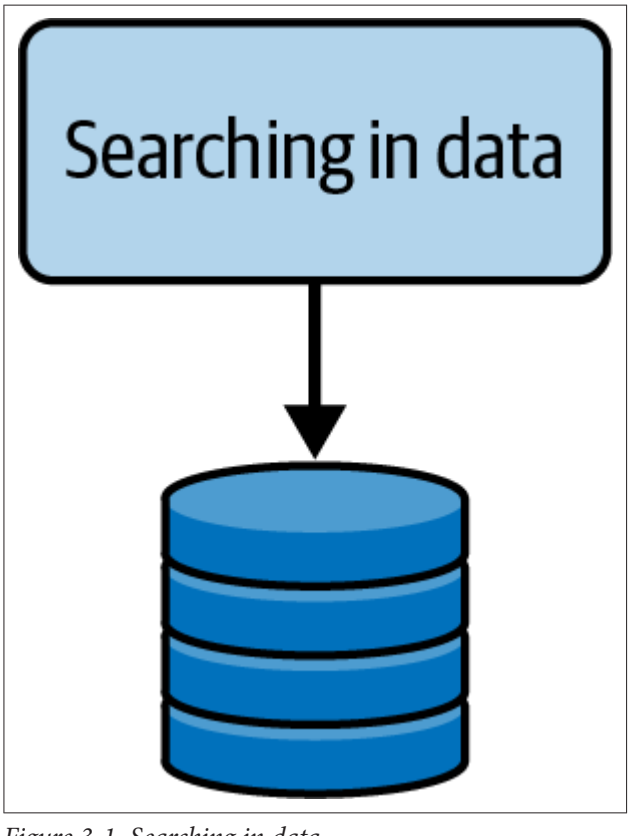


Figure 3-1. Searching in data

The most popular DQL is Structured Query Language (SQL). It allows you to perform queries by writing SQL statements in a DBMS to a relational database. Basically, you can "ask" anything if you master SQL well enough and your relational database holds the relevant kind of data. So, for example, you could ask about customer behavior at certain hours of the day, across a large geographical area, divided into many specific locations. To do this, you would translate what you want to know into an SQL statement, run it in the DBMS, and analyze the results. Further, the results you would get could be turned into business intelligence (BI) dashboards with reports, telling you and business leaders about interesting details on consumer behavior at certain hours of the day across a large geographical area.

In computer science and the data community in general, it is well-known what languages you can use when you search in data, but there is far less attention on the languages you use when you search for data.

Although there's a ton of literature out there for how to use DQLs to search in data, there's actually not much out there in data management literature about how to search for data. The *Data Management Body of Knowledge* (DAMA-DMBOK) mentions that:

> Metadata repositories must have a front-end application that supports the search-and-retrieval functionality required.[1]

And that's it. That's the one(!) sentence on searching *for* data, in a data catalog,[2] in the entire DAMA-DMBOK, the go-to literature for data management. No search functionality is discussed, no techniques are displayed.

Instead of relying on only data management literature, we can broaden our horizons and consider guidance from another area of study—library and information science (LIS). LIS has thoroughly studied searching for data for ages.[3]

In LIS, you will find that there is a parallel dimension of query languages, next to DQL. It's a dimension of query languages that is completely missing in data management literature, as these languages have not been created by data managers and scientists. These languages are known as information retrieval query languages (IRQLs). IRQL has been developed by librarians, archivists, records managers, and most importantly: not data scientists, but information scientists.

But what's the role of an IRQL? IRQL allows all kinds of searches, from very simple to very complex searches. But that sounds just like DQLs, right? You're correct: IRQL does the same thing as DQL. The difference lies in what data layer the languages are applied on: you use DQLs to search *in* data, and you use IRQLs to search *for* data.

To match DQL and IRQL with databases, LIS operates with two kinds of databases:

- Source databases that hold data
- Reference databases that hold metadata about data stored elsewhere

Chowdhury writes that the fundamental difference between the two kinds of databases are that:

---

1 Mark Mosley et al. (eds.), *DAMA-DMBOK: Data Management Body of Knowledge* (Vancouver, WA: DAMA International, 2010), p. 440.

2 Data catalogs are discussed together with other tools grouped as "metadata" repositories.

3 The most complete overview of studies in search is found in Jutta Haider and Olof Sundin, *Invisible Search and Online Search Engines: The Ubiquity of Search in Everyday Life* (London: Routledge, 2019); see especially chap. 2: "Perspectives on Search."

> Reference databases lead the users to the source of information. … Source databases provide the answer with no need for the user to refer elsewhere.[4]

A reference database is a concept, not a specific technology. In Figure 3-2, a reference database holding metadata is illustrated at the left. Once you have searched *for* data and found it, in a data catalog, you are referred to the database in question, and you can continue your search activities there, searching *in* data.

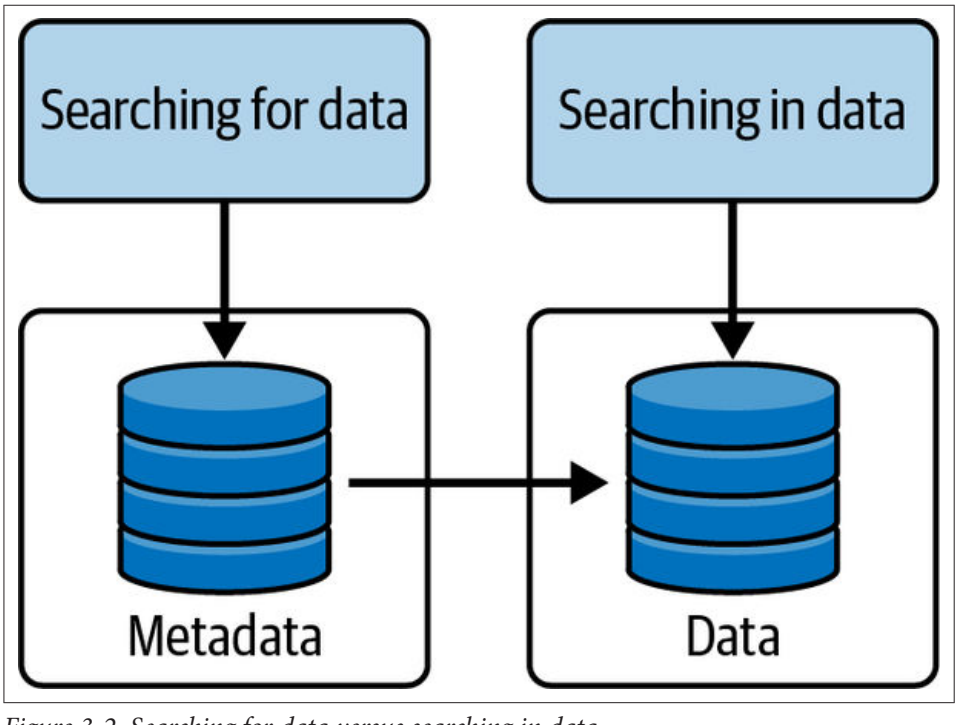


Figure 3-2. Searching for data versus searching in data

Traditionally, information science deals with three kinds of reference databases, namely:

- *Bibliographic databases.* Lists of books on a given topic.
- *Catalog databases.* The collection of books of one or more libraries.
- *Referral databases.* Persons, companies, technologies, etc. on a given topic.

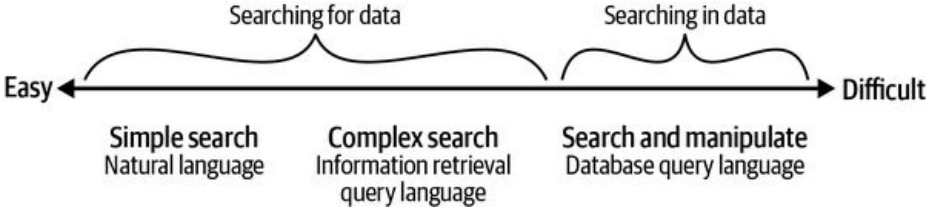
I would add a fourth reference database to this list:

4 G. G. Chowdhury, *Introduction to Modern Information Retrieval* (New York: Neal-Schuman Publishers, 2010), p. 17.

The data catalog. The collection of data in an enterprise.

So there you have it. The data catalog is a reference database. To search it, you can't use DQL, because you are not searching *in* data. Instead, you can use IRQL, the query languages in scope for reference databases that allow you to search *for* and locate the data so you can use it.

Furthermore, searching for data and searching in data can be defined as a spectrum of search, as seen in Figure 3-3. This spectrum goes from very simple searches to search procedures that are very difficult to conduct.



The spectrum of search

The Spectrum of Search still holds true, but AI has changed search for the better, as you can see in Figure 3-3.

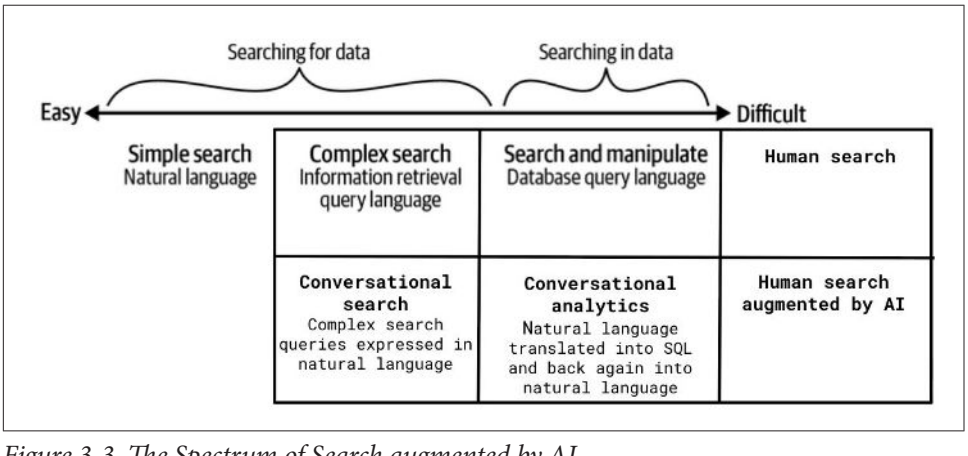


Figure 3-3. The Spectrum of Search augmented by AI

What has changed?

First and foremost, thanks to AI, you are now - to a certain extent - capable of searching for complex data topics in natural language, that will automatically translate into an IRQL search string behind the scenes. You will not see it, and you no longer need to write it. But, it can be helpful to know what is going on. Search semantics is less

explicit, but it still plays a role for finding the data you are searching for. We will cover this in Chapter 4.

Secondly, you are also able to continue your search *in* data, in natural language. AI Analysts are capable of searching in data, by translating your natural language questions into SQL statements and, once you have let the AI analyst search in the data, then, the findings can be translated back into natural language, human readable findings and even dashboards. We will discuss this in depth in Part 2 of the book.

Information Needs: Search Like Librarians—Not Like Data Scientists

Let's look a bit more into what searching for data is all about.

Data scientists excel at analyzing data—from small to massive datasets, they have the tools and mindset to search in the data to extract the findings they need. That's their superpower. Searching for the data for them to work on, however, can be a real challenge because the skills that make them very good at searching in data don't necessarily apply to searching for data. But, as you just learned, there's a significant difference between searching for data versus searching in data.

Librarians, on the other hand, are very good at locating all kinds of material under the sun— books, periodicals, papers, everything! If you ask for it, they could probably search for it and find it. And their superpowers also include searching for data and knowing what data you need.

Unlike data science, which has gained importance only in the last decades, library and information science (LIS) has existed for hundreds of years, with as many years perfecting the art of organizing knowledge and searching for it.

To search as a librarian means, first and foremost, being good at assessing an information need, because the information need determines how you search for data. *Information need* was coined as a term in 1962 by Robert S. Taylor as the way we ask questions to reference databases.[5]

Your information need can be big or small. Ask yourself whether you are searching for:

- Everything
- A few good things
- The sole right thing

5 Robert S. Taylor, "The Process of Asking Questions," *American Documentation* 13, no. 4 (October 1962): 391–96.

- A thing you need again[6]

Your information need determines how you search for data, as needs express different sizes and intentions.

Everything entails complex search—and you're aiming for high recall at the expense of precision. But you can do this in many different ways, as you will see later in this chapter.

A few good things is an information need that is not that clear cut. You can be searching for either relatively high recall or relatively high precision, but not both.

The sole right thing, as the name alludes, aims at finding just one asset, or one precisely defined set of assets. Therefore, these searches strive for precision.

Finally, *a thing you need again* relies on assets that you already know. It also strives for precision, but it's less difficult to search for than *the sole right thing*.

What you need to keep in mind is that searching for data in metadata repositories like a data catalog can be a long process (just as Taylor already pointed out in 1962). So don't get impatient. We are used to being able to google everything, but that's simple search playing tricks on you. Searching for data is not always like that; it can take many steps before you find what you are searching for, and that's OK. You may need to make adjustments to searches, both of the translations of what is intended with a search into how the specific IRQL of the system works and, from that point on, what terms are included, which of those are subsequently excluded or modified, and so on, in a multiple-step long search for the most relevant and valuable hits. This means that search is not just a matter of translating *one* information need into *one* search. The process is way more subtle, and it takes experience.

Searching for data is like driving a car. Sometimes, you're just out for a short ride to pick up stuff. Occasionally, you're on a long, tiresome road on a straight line. Once in a while, you cross mountains with endless sharp corners, the feeling of vertigo, up and down, until you reach the destination. Or it happens that you go one place only to discover that you need to go to a second place, then a third place, before you can return home with the things you were out to get. Sometimes you find yourself in areas with no rules, sometimes the traffic is overwhelming, sometimes the roads are old and full of holes. And sometimes, you just go as fast as you possibly can because it's fun.

Librarians combine all sorts of methods and techniques when they search—you should, too. The next chapter will go over some of the most commonly applied search

6 Information needs can be grouped differently; this grouping is from Louis Rosenfeld et al., *Information Architecture: For the Web and Beyond* (Sebastopol, CA: O'Reilly, 2015), p. 45.

patterns. Although they each can work fine in isolation, they work best when combined.

Serendipity

Serendipity is when you find good stuff you weren't looking for. If you have ever wondered what Google's "I'm Feeling Lucky" button is all about: it's serendipity. Go to Google. Type in a search term and then click the "I'm Feeling Lucky" button. It'll automatically take you to one of the top search results for that term. It might be exactly what you're looking for, but it might not. It might lead you down a different path to get to an answer other than the one you expected. You might even discover new things along the way. That's serendipity.

Compared to just browsing search results, serendipity is something that can be built into the ranking mechanism. At its best, serendipity distracts the user with what they weren't searching for, but what they would be interested in anyway.

Data catalogs, too, should offer serendipity. Here, serendipity is finding potentially interesting assets in unplanned ways. Serendipity is a key enabler for the high usage of your data catalog: users will be naturally drawn to the data catalog if they know they will discover surprising and useful assets when they search in it. Think of serendipity as a magnetic force: the stronger it is, the more it pulls users into the data catalog and the more value it generates.

Serendipity consists of four elements: *insight*, *experience*, *luck,* and *coincidence*. You simply need to maximize each to maximize serendipity, so expressed as a simple formula, it look like this:

Serendipity: insight + experience + luck + coincidence[7]

The more your data catalog takes into account these four elements, the higher the serendipity. The data sources and how their assets are described and tagged must reflect the *insight* of your users, in the sense of what the users know, to appeal to them. Also, the *experience* of your users must be put in play: in this case, the users may not necessarily have great insight about an asset but simply know by experience that it exists—and so being exposed to that creates curiosity. A user—say from HR—may discover

7 The "formula" for serendipity has been put forward as a conceptual way to understand the elements that constitute serendipity in the Danish science forum Videnskab.dk (website in Danish). For further reading on serendipity, see Robert K. Merton and Elinor Barber, *The Travels and Adventures of Serendipity: A Study in Sociological Semantics and the Sociology of Science* (Princeton, NJ: Princeton University Press, 2004).

an asset that describes some experiments from the R&D department that the user has heard of, and confronted with that asset, the user could be tempted to take a closer look at it. So, experience means that search results should also take into account not only what people know due to their field of expertise, but the haphazard experience they gather by being surrounded by other coworkers in a given context.

Finally, the search mechanisms inside the data catalog must take into account *luck* and *coincidence*—luck is not the same as coincidence! Luck is, for example, when a query that the user is not sure will work happens to return useful assets. Coincidence is when the user discovers something that was out of scope of what the user was searching for. You could argue that serendipity is simply coincidence, but it's not; the other elements—insight, experience, and luck—play a role in increasing serendipity to its maximum performance.

You can encourage the likelihood of serendipity by fine-tuning glossaries and descriptions of assets. You can further improve it by using certain mechanisms, such as having the data catalog remember your search history—that would result in a mechanism providing more search results of potential relevance to you.

Serendipity has an evil twin: zemblanity. It's when you find bad stuff you absolutely weren't looking for. Social media is full of zemblanity, of evil, unnecessary comments about—and actions against—other people. Your data catalog will not suffer substantially from zemblanity, as it is a professional tool and people accordingly keep a friendly tone—most of the time. But it can occur, since a data catalog is a collaborative platform that normally offers debate about the quality and potential use of assets. Zemblanity in a data catalog is when you would, for example, stumble upon a vicious, unjust comment about yourself or a close colleague, say in a case where your asset descriptions were misinterpreted to suggest that your work was poor or even amateurish. Watch out for zemblanity—keep it out of your data catalog at all costs!

Promptism

With the introduction of Large Language Models (LLMs), as enablers of conversational search, a new reality has introduced itself to us. As mentioned in the opening of this book: it's a new day for search.

AI visionary and technology entrepreneur Yann LeCun stated that in our era, our eyes have to unlearn to examine what looks like human expressions as human expressions. Instead, what they are seeing is something else, created by machines. With LeCun, I suggest that we think of this in terms of what is to come. We are the last humans with eyes like this. We are the last humans to remember this reality. Future generations will not understand us, but it is our responsibility to carry the legacy of

what our reality felt like, to future generations. We have to tell them that we trusted our eyes in a way that is now forever gone.

I am including Sune Selbæk-Reitz' concept *promptism* here, alongside well established scientific concepts such as *information need* and *serendipity*. Why? Because *promptism* encapsulates the essence of the medium of our time, the replacement of search engines with the conversational machines, some call answer engines. We have to understand that the conversation that is now stimulating our intellectual curiosity with words from the web, is in fact not a conversation. It is something else, created by machines.

The definition of promptism is:

> The uncritical belief that a well-phrased question to an AI will yield a reliable, objective answer; the habit of treating machine-generated responses as truth without examining their source, context, or intention. Often rooted in a misplaced trust in data neutrality and algorithmic authority. The quiet collapse of interpretation into convenience.[8]

Accordingly, a new paradigm of search must take the reality that the concept of promptism addresses, seriously. Just like in the case of zemblanity, the negative consequences of our technological reality of conversational search are less explicit in an enterprise setting than on the open web. We may however still expect that conversational search is not pointing us towards truth, but the feeling of truth.

Accordingly, searching conversationally is a craft that goes beyond collecting good prompts. It's understanding the technology we are using, to not point us towards a feeling of truth, but the data we are searching for. Examples of this will follow in Chapter 4.

Features: Search Features in a Data Catalog

At this point, let's explore how these underlying concepts shape the specific search features in a data catalog.

Basically, you can search a data catalog by simple search, browse, glossary, and advanced search, typically expressed conversationally. All vendors will have these features, but they look slightly different in all data catalogs. Figure 3-5 shows the Hugin & Munin data catalog's search features. By typing directly in the search bar, you perform a simple search; by clicking the Advanced button, you get the option to do an advanced search; click the magnifying glass and you can browse; and finally, if you click the pile of books icon, you search only the glossaries.

8 Sune Selbæk-Reitz: *Promptism - Fluent Machines, Forgotten Questions, and the Fight for Meaning in the Age of AI* (Sedona, Arizona: Technics Publications, 2026), p. 4

Remember that this is the UI of the data catalog in Hugin & Munin. Your data catalog's UI will look a little different, but the search features will be there.

Figure 3-4. Typical Search Features in a data catalog

Now that you know what you can search for in a data catalog and why, it's time to discuss how to actually do the search. Before we jump in, just a quick word about search. When you search, you need to be mindful of *Syntax* and *Semantics*.

Syntax is the specific meaning of an element in your search. When you search, you need to know the exact way to express a value. Be it the spelling of a glossary term, the precise format of, e.g., the serial numbers of your company product, or whatever, you have to apply correct syntax; if not, your search simply won't work.

Semantics is the overall meaning of your query statement. Does what you have written actually search for what you want to find? Are you lacking elements—could some be left out or do they negatively impact the search? Have you mixed up the logic of your search somehow with your operators? If you do not have correct semantics, your search will still run, but the result will be wrong. Your search hits won't be what you are actually looking for.

Keep in mind also that search can only be performed on what is in the data catalog. If data sources are poorly tagged with glossary terms, have no people associated with them, and so on, then the searches performed may be of good quality but will not provide useful search hits. How you organize data defines how you can search it.

In this section, we will go through simple search, browse, complex/advanced search, and thanks to AI; conversational search. Let's look at them one at a time, starting with the one that is the easiest to use.

Simple Search

The first thing you need to know about simple search is that it really is simple: all you need to do is type a word or two in the search bar in order to get results. For example, you can search for `data analytics` in the Hugin & Munin data catalog, shown in Figure 3-6, and then you can expect to get a list of good hits with the most precise one on top. Just like that—simple!

The second thing you need to know about simple search is that it is not simple. The reason why it is not simple is because the algorithm performing the search is doing calculations behind the scenes. In Figure 3-6, the data catalog has autocompleted the word that is being typed, and even proposes several alternatives. We will discuss this further.

Figure 3-5. Simple search for data analytics

Once the search is launched, another set of calculations takes place: namely, what hits will be returned and in what order they will appear in the search result. Again, this is in no way a simple procedure, but a result of the data catalog matching several properties of the query with the crawled IT landscape, along with exactly who you are (the data catalog remembers you).

In the following, you will see examples of search that may appear a bit archaic. That's because AI is changing our search habits fast, and we are all unlearning the ability to use search operators and formulaic logic, when we search. And - that is only a good thing! However, you cannot yet fully count on AI to do the work for you, yet, so you need a bit of introduction to ways of searching for data that will most likely disappear in the decades to come.

In simple search, you can also perform queries of slightly higher complexity, adding more values by combining them with operators. You can, for example, search for sensitive assets in an HR domain (defined as capability in this case), writing the following query, also shown in Figure 3-7:

```
Capability: HR AND ClassificationOfSensitivity: Sensitive
```

Also, data catalogs can have a special feature that enables simple search only in the glossaries. It's accessed by the pile of books icon on the right side of the search bar. This search feature works just like simple search, except it only searches for glossary terms.

Figure 3-6. Searching for sensitive data from HR

Now, let's look at how simple search actually works.

Simple search is programmed to help you behind the scenes. To work at maximum capacity, simple search is powered by a wide range of mathematical procedures. You will never discover how these procedures work as an end user, as it will be part of the intellectual property of the data catalog provider. Nevertheless, you would immediately discover the absence of those mathematical procedures. Without the things going on behind the scenes, simple search would give you nothing but a big mess of meaningless noise, with completely irrelevant search results regardless of what you were searching for. Many of your attempts to do simple search wouldn't even result in hits if you, for example, misspelled a glossary term or couldn't remember an old project name and guessed wrong—even if you came close.

A well-programmed simple search will subtly help you by correcting your queries, suggesting other queries, and even remembering your search habits. Some of the technology features that do this are:

- Autocomplete

- String matching
- Synonym ring
- Thesaurus
- Ontology
- Search behavior

Although these features work behind the scenes, you need to know them, because you can influence them and thereby improve your simple search.

Autocomplete is a live suggestion for endings of words as you type them, just like you see in Figure 3-6. It draws on the glossaries in the data catalog. So the more you enrich your glossaries and the more you search, the better predictions you get. It can also be enriched with natural language processing and machine learning.

String matching (also referred to as *fuzzy logic*) searches for all the values that come closest to the search that is being performed. String matching not only deals with misspellings, it can also take into account the many alternative versions of acknowledged ways of writing, for example, names (e.g., Dostoevsky/Dostoievsky), as well as conventions for digits (e.g., different ways of writing dates, such as 20-05-2022 and May 20th 2022) and acronyms (e.g., NATO/OTAN) and similar.

Synonym ring refers to a group of data elements that are considered semantically equivalent for the purposes of information retrieval. You might have seen this when shopping online and the website suggests other products that might interest you. A ring of synonyms can also be relevant in a data catalog. It could be in cases where projects are renamed or when they are reignited or refocused. In global companies with product names that vary between sales regions, such as is the case in pharma, a synonym ring will also be very useful in a data catalog simple search, just like product alternatives on the open web.

Thesaurus is the broadening of synonym rings to the global glossary, as discussed in Chapter 2. It's the entire cluster of glossary terms surrounding the value you searched for, which will affect the selection and ranking of hits in your search result.

Ontology: if your data catalog is based on a knowledge graph, then nodes close to the node you are searching for would be ranked high in the search result, also.

The better you build your thesaurus, and the more you use it to organize your data, the better a simple search feature you get.

Search behavior is when the data catalog simple search remembers your search habits (what you searched, what you ultimately selected) and those results influence your later search results. In the best data catalogs, your search behavior is taken into account by the algorithm performing the search when your search results are selected and ranked.

Another way to search the data catalog for the information you need is by browsing.

Browsing

We've all browsed through shelves with books, collections of old-school vinyls, or through massive amounts of pictures in Instagram. You don't really know when to stop, and you might not be entirely sure what you are searching for. Browsing is when you're scanning through content without meticulously looking at all the details of each element in that flow.

For the sake of simplicity, more than technical overengineering, I suggest you think of browsing in three dimensions:

- Vertical
- Horizontal
- Graphical

In the data catalog in Hugin & Munin, you browse by clicking the magnifying glass. Under the magnifying glass, your domain opens downward in the *vertical* structure of domains, in more and more specific subdomains. You can always browse up again, and then down again. Below the most specific subdomains are your data sources and finally your assets (see Figure 3-8), from `Financing`, to `Customer Management`, where you can find data sources with data assets.

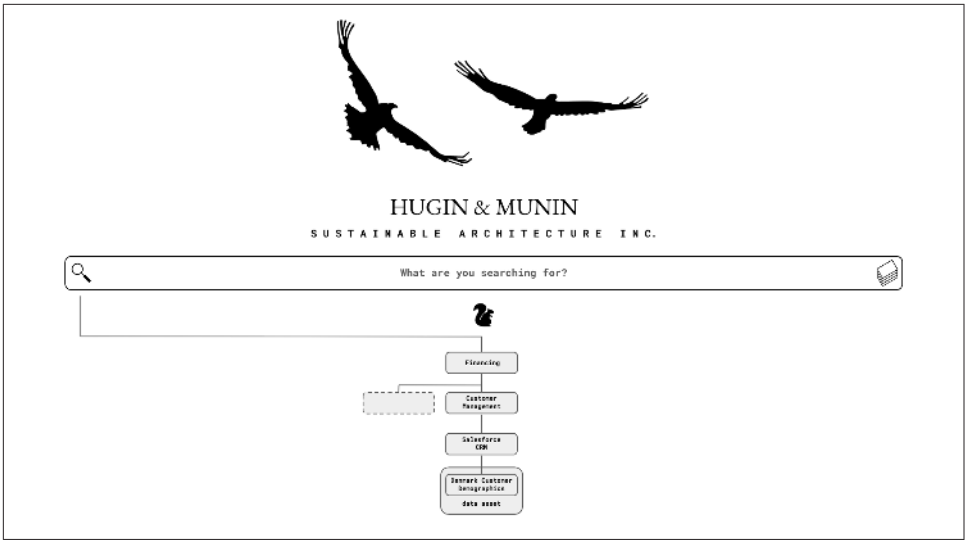


Figure 3-7. Browsing vertically in domains

From this point on, you can browse sideways - *horizontal* - in data lineage. This enables you to see how your data has traveled from the source upstream, and how it continues further downstream, as depicted in Figure 3-9.

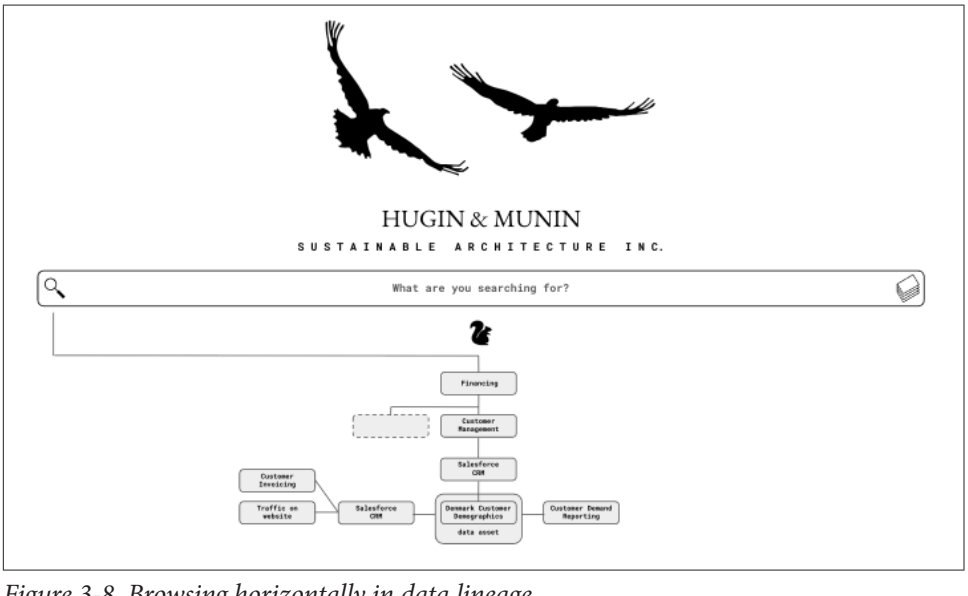


Figure 3-8. Browsing horizontally in data lineage

There are at least 14 different ways to approach data lineage,[9] and you should not expect all those to be present in one solution—not even close. More, you should approach data lineage as a mindset - a mindset you decide what should depict.[10] As you will see in Chapters 4 and 6, data lineage is a very useful feature for both governance and analytics stakeholders.

You can also browse *graphically*. By doing that, you move associatively from your asset toward other assets that hold data related to some of the data in the asset that was your point of departure. You can see this illustrated in Figure 3-10.

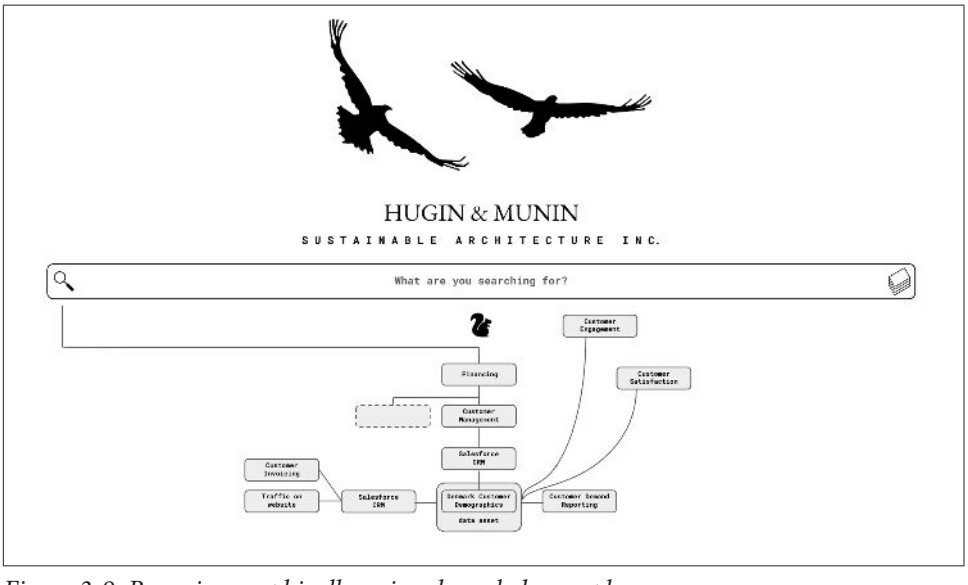


Figure 3-9. Browsing graphically - via a knowledge graph

Knowledge graphs power search engines on the web, but knowledge graphs also go back several decades as technology.[11] The reason why it is so powerful is that it allows you to relate all your assets as you want and search for them as you please.[12] The latter

9 Bonnie K. O'Neil and Lowell Fryman, *The Data Catalog: Sherlock Holmes Data Sleuthing for Analytics* (Basking Ridge, NJ: Technics Publications, 2020), 196–97. Albeit in my view, of these 14, some must be regarded as semantic relations or knowledge graph relations.

10 A very useful guidance on data lineage is found in; Irina Steenbeek: *Data Lineage from a Business Perspective* (Data Crossroads, 2021), p. 77-81

11 Claudio Gutierrez and Juan F. Sequeda, "Knowledge Graphs", *Communications of the ACM* 64, no. 3 (March 2021): 96–104.

12 Mark Needham and Amy E. Hodler, *Graph Algorithms: Practical Examples in Apache Spark and Neo4j* (Sebastopol, CA: O'Reilly, 2019).

is the most desirable for your company, as it best captures the vastness of data and the connections between it.

If you have organized your data perfectly vertically, horizontally, and relationally, you have the possibility to browse that organization again in all three directions. You can browse your data landscape horizontally in domains and subdomains. At any given point, you can continue browsing vertically, following the lineage of a given asset in a domain upstream or downstream, outside of the domain where the asset is placed. Or, if you want, you can continue to browse relationally, in elements that are associatively connected to your asset.

As you can see, if you organize your data to the most complete level in the data catalog, what you get is an easy, seamless flow in your browsing experience across all the data in all directions in your entire IT landscape.

Browse search puts you in a position of perfect discovery for things you were not necessarily looking for but which could be very useful now that you know that they exist… Serendipity!

And now, we will look at the kinds of search that require the most from your end users' side but will deliver fantastic results if they master it.

Complex Search

The first thing you need to know about complex search is that it really is complex. To truly master complex search, you must be cautious about two elements:

- You must carefully use the correct syntax in the query language—if you misspell your operators or values, then your query won't run; and
- Even more difficult, you must be very aware of the semantics in your search. Are the assets in your statement actually the ones you are looking for? Did you reverse the logic of the Boolean operators? Was your grouping of values correct? You need to be sure of that.

That said, don't be afraid to fire off wrong queries; just pay attention to what happens, analyze the result, and eventually adjust and search again. Unlike when searching in data, searching for data with wrong, heavy queries has no computational cost.

The second thing you need to know is that complex search is in fact very simple. Not for the user, but for the data catalog as software. It is easy to deliver a complex search capability from the data catalog provider's side—a substantially smaller part of the technologies behind simple search are necessary to perform complex search.

There is a yin-yang relationship between simplicity and complexity in search, a *complexity of simplicity and simplicity of complexity*. Simple search may appear very easy to you, but it takes computational efforts. Complex search may, on the other hand, appear complex to you, but that is because there has been only a little computational effort.

Figure 3-10 shows a complex search. The search combines one or more asset stewards with one or more generic data sources and with one or more glossary terms from taxonomies. Finally, pay close attention to the use of the asterisk (*), which allows multiple endings.

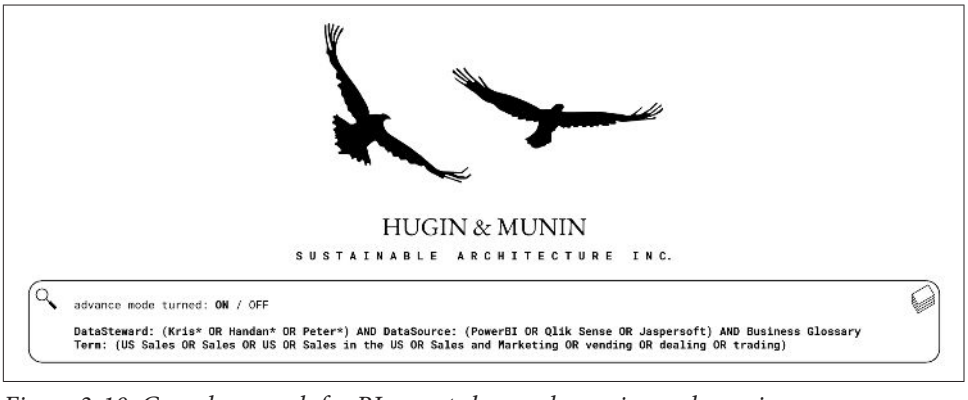


Figure 3-10. Complex search for BI reports by employees in a sales region.

You may ask yourself: "If complex search is so hard for me as an end user, compared to simple search, why should I do it at all? Why not just stick to simple search?" It's because sometimes simple search simply doesn't give you the answers you want, in relation to what you are searching for. We will discuss this more in Chapter 4.

With complex search, what you see is what you get. There is nothing going on behind the scenes, as in simple search, or as you will see in a minute, in conversational search, . You have to write the entire search query, and you have to get syntax and semantics right. Accordingly, building complex search relies on technologies at a very low level of complexity. The data catalog has to understand and execute the operators on the values you apply in the search—that's it.

Complex search will typically yield a long list of hits, and you will need to peruse the results and assess if any of them fit your needs. You shouldn't skim through the results and focus on just the ones that happen to catch your eye, because you might miss something; that is *browsing*. Instead, you will be *perusing* the search results. *Perusing* is to read something in a thorough or careful way. In the most intense hunts after data in complex search, you will be perusing the search results returned.

Do you find complex search to… complex? Fear not. This is finally changing. AI is transforming how we search - also in a data catalog. Welcome to a new era, where the way you search mirrors how you speak.

Conversational Search

As mentioned in the preface, Microsoft CEO Satya Nadella, when announcing the partnership with Open AI, said:

"It's a new day for search"

He was referring to the end of the search engine era, as conversational search was taking over, thanks to generative AI, powered by Large Language Models. But - what happened was not a replacement: Conversational search is not a faster way to perform what the search engine does. Conversational search changed the very nature of search. Complex topics - that were unthinkable to phrase in the keyword language of the search engine - were finally made possible.

Instead of search engines being completely superseded,[13] conversational search is simply of a different nature - another dimension of search, where deep context of a complex topic is finally able to be expressed by end users not knowing the given IRQL of the technology they have at their disposal.

Consider the complex search in Figure 3-12, performed in advanced mode with exact syntax, logically structured query semantics and wisely placed operators to cater for alternative spelling of names…. It's not an easy job. And, we have now entered an era, where this is beginning to not be necessary anymore.End users can ask these complex questions in natural language, and enter a conversation. *That* is the new day for search, and it is depicted in Figure 3-11.

13 see e.g. the conclusion in: Tafesse, Wondwesen, and Yoseph Mamo. 2026. "A Comparison of Conversational Chatbots and the Internet for Consumer Information Search." *Behaviour & Information Technology* 45 (2): 314–31. doi:10.1080/0144929X.2025.2517215.

Figure 3-11. Conversational search, complex search made easy.

Remember. The current AI era is in its infancy. There is a long way to go, don't expect this capability to have fully replaced complex search in your data catalog.

Now, let's look at the mechanics at play when we search.

Mechanics: The Mathematics Behind Search

By now, you know why you search in a data catalog, you know what you are searching for, and you know how to do it. The final section in this chapter is devoted to the mechanics of search. As you are about to discover, the mechanics of search will help you understand when to apply which kind of search for data. You need to take into account the mechanics of search when you are assessing what your information need is and how you should formulate or perform a search to meet that need. In this section we cover:

- Recall and precision
- Zipf's law

Recall and Precision

Imagine that you discover a red area on the upper side of your left hand. It itches, and the skin in that area looks dry. You book an appointment with your doctor. Once you are face-to-face with your doctor, you ask her: "Is this eczema?"

We all intuitively think that two things can happen from this moment on. The doctor can say either "yes" or "no." But in fact, four things can happen because, besides diagnosing correctly, the doctor may diagnose wrongly. The doctor can say "yes" but be wrong, or say "no" and be wrong.

Being diagnosed with a medical condition is referred to as being *positive*. Diagnosed as not having the medical condition is called being *negative*. And accordingly, getting positively diagnosed without having the medical condition is called *false positive*. Likewise, if you have the medical condition but are diagnosed as negative, you are a *false negative*.

Not only is this true in medicine when diagnosing patients, it's also true when calculating the effectiveness of search in data catalogs. Here's how.

In Figure 3-11 you can see what is called a confusion matrix. In it, you can see that the confusion matrix groups *true positives* and *false negatives* as all the actual positives. And it groups the false positives and true negatives as all the actual negatives. Further, just like patients in the case of medicine, you can see that, despite how you search in a data catalog, you will have *true positives*, hits that are relevant, and *false positives*, hits that are not relevant. There will also be a series of hits that are not found in your search, even though they are relevant; these are the *false negatives*. And there will be a remaining group of hits you did not get that are *true negatives*—those are the hits that are correctly not part of your search result.

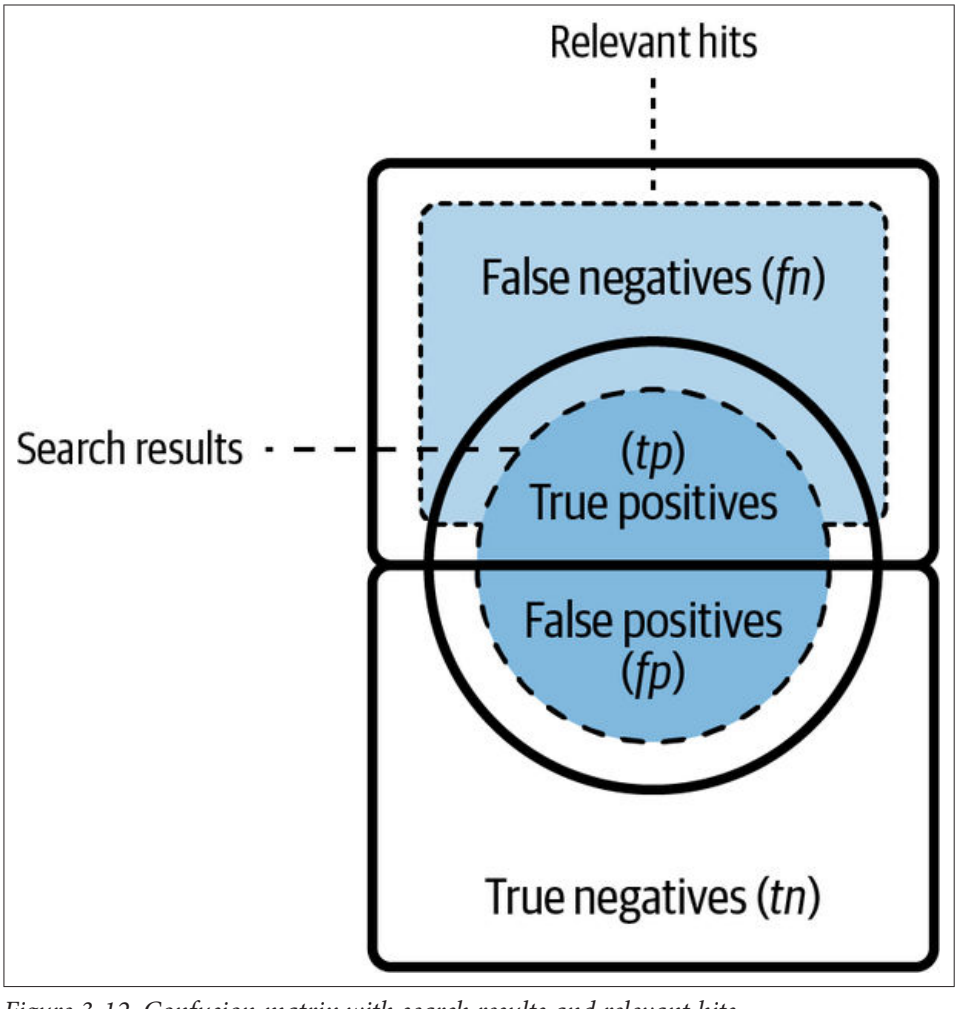


Figure 3-12. Confusion matrix with search results and relevant hits

Recall and precision are the most fundamental mechanics of search. No search you will ever conduct escapes the confusion matrix: some hits will be retrieved without being relevant, and some will not be retrieved, even though they were relevant.

The confusion matrix is used to represent many mathematical formulas that go deep into the inner workings of classification—including for search in a data catalog. You do not need to know all the formulas at play in the data catalog that are based on the

confusion matrix.[14] But you do need to know about *recall* and *precision*, because you must take them into account when you search.

Recall is calculated by dividing the number of true positives (*tp*) by the sum of the number of true positives (*tp*) and the number of false negatives (*fn*). Metrics must be put in place to calculate recall or make an approximate assessment of recall.

$$Recall = tp / tp + fn$$

Recall measures how many relevant hits you retrieve in your search—in relation to the total number of relevant hits in your data catalog. The complex search function ranks recall highest, as you typically want all results of even just potential relevance to what you are looking for when you formulate long, complex queries in your data catalog. Basically, you are looking for assets of just the slightest relevance in your entire data catalog when you are maximizing recall.

Recall is often difficult to calculate because you have only approximate ideas about how many false negatives are actually in your data catalog. Therefore, if you believe that your search results are not providing enough recall, assess a relative number of false negatives and do the calculation. Either you work on improving the ratio to around 0.75 (see Chapter 4, "Flexible Simple Search," on how) or, as an alternative, browse the catalog to evaluate if your assessment is correct.

Precision is calculated by dividing the number of true positives (*tp*) by the sum of the number of true positives (*tp*) and the number of false positives (*fp*).

$$Precision = tp / tp + fp$$

Precision is always easy to calculate, because you only evaluate the search result and not what you have not found. It's therefore also fair to strive for very high precision and aim for 0.9. It's also easy to improve, because you can modify search and easily measure if your precision improves.

The simple search function ranks precision highest, as you typically want just a few, exact hits, which helps you in what you are doing right now. Contrary to recall, you cannot expect to find everything of potential relevance when maximizing precision. Pertinent information in your data catalog is left out when you strive for precision,

14 But a mathematical understanding of these laws can lead to improved search features both in your data catalog and beyond; see G. G. Chowdhury, *Introduction to Modern Information Retrieval* (New York: Neal-Schuman Publishers, 2010), chap. 9. and H. Nelson, *Essential Math for AI* (Sebastopol, CA: O'Reilly, 2023), p. 109-111

and that's OK—you are not looking for each and every potentially relevant search hit in the catalog, just for whatever gets you going with what you want to do.

You need to know recall and precision because you cannot search in a way that maximizes both at the same time. So if you strive for precision in your search, be aware that you will not receive a search result that caters to recall. And vice versa, if you are searching for every potential asset of relevance, you will not get a search result with very high precision.

In "Simple search", I described mechanisms that improve simple search. These mechanisms are all mathematical calculations taking place behind the scenes to make simple search powerful and easy for the end user. Basically, all these mechanisms serve the purpose of precision.

When you're aiming for precision, don't expect recall, and don't try to obtain it. Precision results in one—or a few—relevant search hits, not a big set of potentially relevant search hits.

In "Complex search", I described mechanisms that improve complex search. These mechanisms are not taking place behind the scenes but are performed by the end user, either as a query language or as point-and-click options, combined with drag-and-drop tools to formulate complex queries. Basically, these mechanisms serve the purpose of recall.

When you are doing complex searches, you're aiming for recall. Don't expect precision, and don't try to obtain it. Take a look at Figure 4-5. It's a long statement. It results in many, many hits, of which a lot may be relevant. That's recall.

In Chapter 2, I discussed exhaustivity and specificity. Think of nicely curated assets, with many glossary terms assigned to them from exhaustive glossaries, so that the asset has a high degree of specificity. Now, think of this in relation to recall and precision. Searching for those nicely curated assets is possible by using a high level of exhaustivity in glossaries, which increases recall and decreases precision.

Zipf's Law

There is an inevitable problem built into the type of data catalog that crawls: the more you crawl, the more your crawled metadata loses its meaning; it might be depicting your assets correctly, but only in ways that group more and more different assets with the same kinds of metadata, even though the assets have little or nothing in common. Imagine tables in many different data sources that all have a column called *Efficiency*,

Result, or *Score*. There is nothing wrong with these column names; they depict the values in them. However, the columns in the different tables in the different data sources have absolutely no relation to each other. But, when crawled by the data catalog, they are all part of the search result if you search for, e.g., *Result*, even though they do not have anything in common. The more data sources you crawl, the more this will happen.

The problem is rooted in Zipf's law—you need to know it, to avoid it.

Zipf's law is named after George Kingsley Zipf (1902–1950).[15] The law states that the frequency, *f*, of words are, more or less, inversely proportional to their rank, *r*.

$$Frequency(f) \propto 1 / rank(r)$$

For example, the most common word in English is "the," and it occurs once for every 10 words in a typical text. The second most common word is "of," and it occurs once for every 20 words in a typical text—and so it goes. The bigger the body of text, the more accurate Zipf's law is. Zipf argued that two opposing forces compete in our language: *unification* (general words with many meanings) and *diversification* (specific words with precise meanings).

Zipf's law is also at play in search.[16] .[10] Here, it has been demonstrated that Zipf's unification equals *description*, meaning a complete description of an asset. And diversification equals *discrimination*, the ability to distinguish assets from each other.

The problem is that more and more data is pulled or pushed into the data catalog from table and column names, folders and files. And Zipf's law just gets more and more true the more times the data catalog does this: the number of names that strive toward description, and not diversification, will increase as the number of data sources increases in the data catalog. The same words will inevitably have more and more meanings. And then suddenly you are in a situation where, say, hundreds of tables hold data about *Efficiency*, *Result*, or *Score*. And that may be true; these tables are correctly described with the crawled metadata. But they are not discriminated: you cannot tell one from the other—you don't know which is relevant in what context.

This is where glossary terms come into the picture. To counteract Zipf's law, you simply need to tag your assets with glossary terms to make your assets stand out and be distinguishable from other assets that hold the same kind of crawled metadata even

15 Although he didn't really invent it and it isn't really a law. See Erez Aiden and Jean-Baptiste Michel, *Uncharted: Big Data as a Lens on Human Culture* (New York: Riverhead Books, 2013), 249, Zipf's law.

16 David C. Blair, "The Challenge of Commercial Document Retrieval, Part I", *Information Processing & Management* 38, no. 2 (March 2002): 273–91.

though they stem from different parts of the business and are really about different things.

For example, the above-mentioned tables about *Score* could be tagged with terms such as "wood durability," "customer satisfaction," or "employee performance" so that they become distinguishable from one another.

Summary

In this chapter you have become familiar with search - with its concepts, features, and Mechanics.You need these for understanding the elements at play when searching a data catalog:

- Concepts
  - — *Searching* for *versus searching* in *data*. Searching for data takes place before searching in it. It is really about finding the data sources that you want to use, outside of the reference tool that you are using, in our case a data catalog. When the datasource is found, you can search in data.
  - — *Information needs*. Not every need for information is the same, sometimes you are searching for a broad context, asking for many search results, and sometimes not. The information need impacts how you search for data.
  - — *Serendipity*. Serendipity is the surprising opportunity to find useful data you weren't looking for.
  - — *Promptism*. The concept that describes blind faith in conversational search, instead of healthy skepticism.
  - —
  - — When searching for data, you need to apply the mindset of a librarian, not a data scientist. Searching for data is a discipline that relies on search mechanics, but it also takes experience and understanding your company's data and language.
- Features
  - — Simple search is simple for you, but complex behind the scenes. It provides search results based on how you have previously searched. It also corrects your queries and makes suggestions.
  - — Browse search can be vertical, horizontal, and relational. Vertical browsing is based on domains. Horizontal browsing is based on data lineage and displays how data travels across systems. Relational browsing is based on knowledge graph technology and maps how data is conceptually connected.
  - — Complex search is complex for you but simple behind the scenes. It requires you to master an IRQL and apply it when you search. You need to take into

account syntax and semantics of your searches, so that your hits reflect what you are actually looking for.

  - AI powered conversational search is everything complex search is not: it is easy for the end user to build long conversations that covers multiple topics intrinsically connected to each other,

- Mechanics
  - The mechanics of search describe the most basic mathematics behind search:
  - Recall and precision enable you to measure how well your complex and simple searches work.
  - Zipf's law explains why you can't rely on crawled metadata. The more you crawl, the more your crawled metadata loses meaning.

Next up will be an aha moment: in the following chapter, you'll see search applied for the data in your company.

CHAPTER 4

Search For Data Patterns

A Note for Early Release Readers

With Early Release ebooks, you get books in their earliest form—the author's raw and unedited content as they write—so you can take advantage of these technologies long before the official release of these titles.

This will be the 4th chapter of the final book.

If you'd like to be actively involved in reviewing and commenting on this draft, please reach out to the editor at *shunter@oreilly.com*.

Have you ever noticed that how you search depends on what you're searching for? In this section, I will discuss typical patterns of searching for data. You will see how recall, precision, serendipity, exhaustivity, specificity, promptism, and other concepts are at play when you search.

Have you noticed that you are being addressed as a reader - as a human? Remember that in this chapter, I address you, as a human, and how you search. AI Agents search for data, too. I describe how they search, and will search, in Part II of the book.

Search Pattern Overview

All the search patterns are listed in Table 4-1, with a search name, search type, a short description, and the search's relative level of precision and recall. Note that precision and recall are not applicable to browsing, and that the search types soften the distinction between simple search, complex search and conversational search.

For clarity and consistency, I have kept all the search patterns from the first edition. Therefore, as you will see below, certain patterns seem dated or unrealistic at this point in time. I have kept so that you know, what conversational search is replacing.

Table 4-1. Search patterns

Category	Search name	Search type	Description	Precision	Recall	reconfigured by AI
keyword + conversation	Basic simple search	Simple search	A few precise hits and a lot of noise	High	Low	no
keyword + conversation	Detailed simple search	Simple combinatorial search	Slowly formulated simple search, because the search must be precisely formulated	High	Low	yes
keyword + conversation	Flexible simple search	Simple combinatorial search	Truncated search that eases and broadens a simple search	Low	High	yes
keyword + conversation	Range search	Complex combinatorial search	Range search that allows retrieval of assets between two values	High	Low	yes
keyword + conversation	Block search	Complex combinatorial search	Combination of selected terms to depict a topic	Low	High	yes
keyword + conversation	Statement search	Complex combinatorial search	Long statement of precise conditions assets must meet	High	Low	yes
Browsing	Glossary browsing	Browse search	Lookups after specific word results in lists of words in the glossary that can be browsed	—	—	yes
Browsing	Domain browsing	Browse search	Domains as explained in Chapter 3	—	—	no
Browsing	Lineage browsing	Browse search	Lineage as explained in Chapter 3	—	—	no
Browsing	Graph browsing	Browse search	Graphs as explained in Chapter 3	—	—	no

Let's talk about each search pattern.

Keyword Search and Conversational AI Search

In this first section, we will go back and forth between examples that may seem old school - pre AI - and examples that are almost not yet possible, with AI. That's the moment we find ourselves in, and it's an exciting moment to be in. Because it means that we, you and I, get to explore and define what conversational AI should be all

about. What you see below is only the beginning. Please continue with your own search experiments. We are defining the future of conversational search together.

Basic Simple Search

In general, your casual searches during a typical workday are probably basic simple searches with a couple of words in the search bar. You do these kinds of searches when you are not aiming for anything near total recall—you just want something relevant, instantaneously. What matters in this type of search is the hit at the top of the search results. That hit, seen in isolation from the rest of the search result, must be a perfect precision hit. Anything below that top hit doesn't matter.

Simple searches use plain-language search terms and not query language. *Basic simple search* is the least complicated kind of search you can do, as it consists of only one or two plain-language search terms, such as "good weather" or "summer."

For example, let's say that a sales rep for Hugin & Munin assigned to Sweden wants to find the latest, most relevant sales BI report for their area. They might do a basic simple search for sales, as shown in Figure 4-1. The sales rep, being an average end user, expects the top result to be the exact thing they were looking for. If it's not, then they move on to more-complex search patterns to try to find what they are looking for - in today's era most likely through conversational search. Because simple search takes into account all of the technologies mentioned in Chapter 3, such as prediction, fuzzy logic, and history of search behavior, it does a good job of figuring out what the average user wants.

Figure 4-1. Simple search for `Sales`

The sales rep will expect to get the latest, most relevant sales BI report for the area the user is a sales rep in. This reflects the most precise, relevant hit on top, based on who the user is, what kind of data most interests the user, and how the user has previously searched.

If your data catalog is based on a knowledge graph, expect to have a very powerful simple search feature. Search results will enlighten you as to the business contexts of a given asset and be ranked with high precision. This is, for example, the case with Google's Knowledge Graph.

Basic simple search will be the only way many end users will use the data catalog. This search engine–like experience creates an impression of ambient findability—but that's not what it is. It's the easiest kind of search. It will offer end users precision at the expense of recall. Users will be able to find the one right thing at the top of the search results.

It's common for data catalog providers to demonstrate basic simple search as the only way to search in sales material—it's often this exact way of searching that handles whatever users are searching for. Nevertheless, it's impossible for this kind of search to deliver on all information needs. But other kinds of searches are more difficult and time-consuming, and therefore rarely promoted.

Detailed Simple Search

Sometimes you are looking for one type of thing and only that—and you know how to express it, if you concentrate. This is a relatively simple search that is not fast, because you have to get your search syntax right; it's detailed. You might even have to do a couple of initial searches to test that everything works as intended.

Detailed simple search is when you need to use a bit of query language to formulate your search statement. This search is slow, because it relies on users to type exact values, which requires attention, and this slows down the search process. The search type is a simple combinatorial search, because it's a relatively simple search statement that is combined with only one Boolean operator. If you take a look at the spectrum of search in Figure 3-3, we are moving away from *easy* toward *difficult*.

In Hugin & Munin, our fictional sustainable architecture company, end users make use of their well-curated glossary, which allows them to search for finely granulated words, e.g., for types of wood: heartwood, spruce, pine, and so on. Words for wood in the glossary are the standard English names for kinds of wood combined with their Latin name. Let's say you want to search for assets with the steward John Miller that hold data about ash trees from the global glossary, like this:

```
GlobalGlossary:Ash Fraxinus AND DataSteward:John Miller
```

It's possible to type this in the simple search view without completely losing the overview of the search typed inside the simple search bar, as in Figure 4-2.

Figure 4-2. Detailed simple search

This search gathers all the data assets/data products with the global glossary term `Ash Fraxinus` that have `John Miller` as an steward. In this search, the user would perhaps need to determine the right way to express the type of wood in the global glossary before performing the search. This search will take some time to build but will deliver precise results, as only the assets with the distinct characteristics of the search are returned. So, unlike *basic simple search*, all hits are relevant here, precision is high and recall is low, and the search itself takes a little time to create.

Let's be honest about it: Obviously, this type of detailed simple searching may already seem archaic. A conversational search style is emerging that is (or will in the years to come) capable of replacing it, as seen in Figure 4-3.

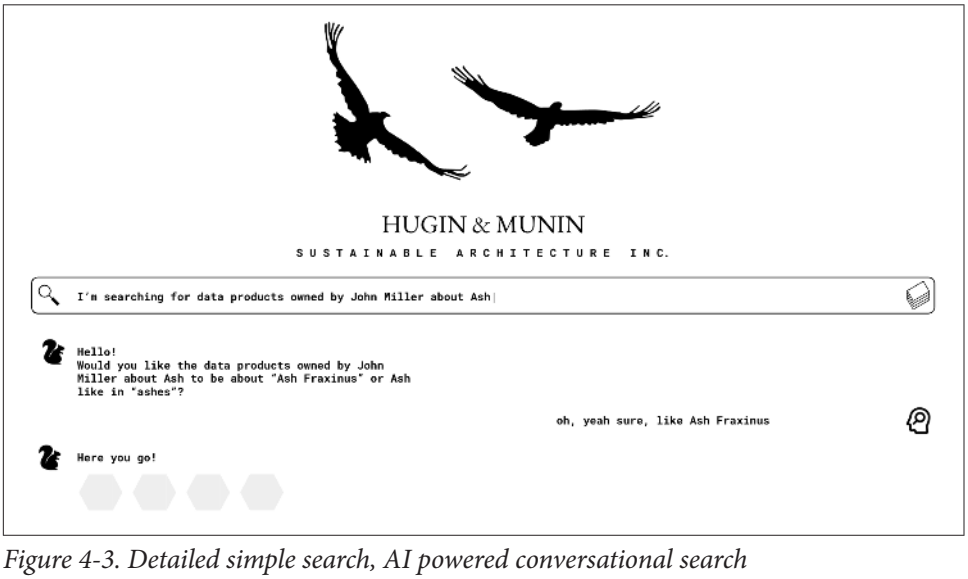


Figure 4-3. Detailed simple search, AI powered conversational search

Ideally, the conversational search will be able to filter the information need such that the boolean operators are correctly applied, and the semantic intention is captured.

Besides having to accept learning how to search archaic systems that have not developed their search features enough, you are also reading this chapter - with these somewhat old school examples - so that you can prompt with precision, when you are searching for data conversationally. That takes understanding of the logic that is underneath the conversation.

You can also loosen the syntax and move away from a detailed simple search pattern into a flexible search pattern. In that case, simple searches are not totally precise but become relatively fast.

Flexible Simple Search

You may also sometimes need to perform searches that are imprecise and that will require some perusing through the search results to find the assets you had in mind.

That's *flexible simple search*, and it's a little faster to write than detailed simple search because it depends less on exact syntax; you don't need to know the exact values in your query statement. Flexible simple search is also a simple combinatorial search, but it allows for a larger set of search hits and a higher recall, at the expense of precision.

For example, a group of Hugin & Munin employees in the communications department need to know what kinds of wood the company uses in order to include some details in a press release. They heard that the info is in a CSV file. They don't know how the asset is described in the catalog except that it contains data on wood and it's a CSV file. They might search the following, shown in Figure 4-4:

Figure 4-4. Flexible simple search

This search results in all assets that represent CSV files and that have folksonomy terms with the word `wood` in them, but truncated on both sides so that the results are open to all combinations with wood. So, for example, free glossary terms such as `wooden floor`, `beautiful wood`, `woods`, and so on are automatically included in the search.

This type of search will provide high recall and therefore compromise on precision. And that's the point: the end user does not know how to search this in a way that delivers complete precision and must therefore aim for higher recall to retrieve a group of assets wherein the asset is located.

Could this search search be performed with AI, conversationally? Yes, of course. However, in that case you would expect a set of search results, as is currently provided in conversational search - and as you are searching for all relevant results, this search is kept as a classical keyword search with operators.

Range Search

Sometimes you have to search for something between two points, such as dates or anything that holds organizational logic in serial numbers.

That's done with *range search*. It's a more refined complex combinatorial search type, which uses one or more Boolean operators and at least two values that establish a range.

For example, if you were looking for a given set of hypotheses that were tested sometime around when particular projects were carried out, you might search research projects like this:

```
> RES.100.7.1003 AND < RES.100.7.1837
```

It can also be room numbers on floor plans, equipment, and so on.

For example, a project team in Hugin & Munin wants to analyze all pictures of heartwood between November 2012 and February 2018. They search like this, shown in Figure 4-5:

```
Unstructured Data product:Pictures of heartwood AND (< 01.31.2018 AND > 10.31.2012)
```

That search returns all hits that refer to pictures of heartwood in the specified period of time.

Figure 4-5. Range search

Again, with conversational search, this can be turned into a smoother, simpler question, like in Figure 4-6.

Figure 4-6. Range search, AI powered conversational search

Block Search

Say an unhappy customer has decided to file a lawsuit against Hugin & Munin. The house Hugin & Munin built for him has cracks in the facade, and the customer argues the wood that the house is built of is not solid enough.

Block search is a very comprehensive complex combinatorial search, where you are searching for an entire topic. Generally, a lot of different things and words are at play

in such a search, and you order those in related groups as blocks, hence the name *block search*.

The lawyers in Hugin & Munin start their due diligence by searching. Using their basic training, they search the data catalog for reports and test data that examine the hardiness of different kinds of wood in the company's own constructions. They combine a large selection of words to maximize recall—they have to get every single potentially relevant asset, with the consequence of having little precision, so they expect to be perusing quite a lot through the search results. They search as follows, shown in Figure 4-7.

```
(DomainTerm:((Pine OR Ash OR Beech OR Oak OR Wood) NOT Linden) OR FreeTerm:Wood OR GlobalTerm:(Pine
```

Figure 4-7. Block search

The search results will show assets that have one or more combinations of terms for *wood* and terms for *hardiness*.

The search consists of domain glossary terms from different domains that describe types of wood in standard English. The assets must have one or more of these words, unless the asset holds the next value, the free glossary term *wood*, or one or more of the global glossary terms for wood—as the word `wood` is not a term found in the global glossary. It can also hold a mix of these words. If one or more of all these criteria are met, then these must be matched with the domain glossary terms for hardiness.

But not all lawyers have been trained in the data catalog's query language, and they get a little dizzy trying to control the syntax while at the same time focusing on the semantics. Therefore, some of the lawyers just use the search builder. They enter the search builder from the advanced search field. The search builder allows end users to formulate their search with point-and-click options, which removes the stress of

checking the syntax and focuses only on the semantics. You can see the search builder in Figure 4-8.

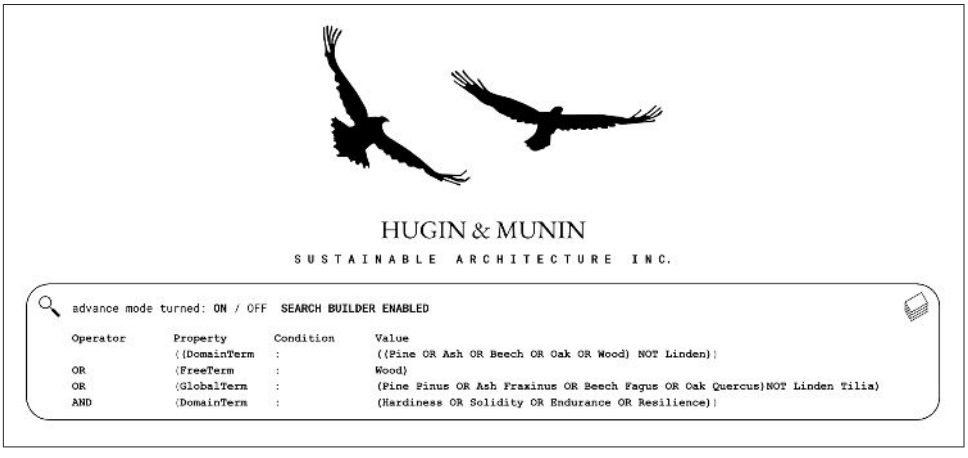


Figure 4-8. Search builder enabled

The search builder in Hugin & Munin creates a visually navigable overview of long searches. All the parentheses that are gray are optional; they become active if the end user clicks them. This way of cutting up the complex search makes it easier to keep an overview of what the search does, so that the semantics especially are easy to keep an eye on.

Search builders like the one in Figure 4-8 are standard components in reference databases such as PubMed. Many data catalogs also have a search builder.

This type of search is also called block search in LIS. It's practiced as a method to obtain large sets of search results for complex searches. Normally, this kind of search has several phases, where words are added, others removed, in a series of adjustments that make the searcher capable of translating what data is needed to the language and structure of the data catalog, based on analysis of the hits retrieved from the previous steps in the search.

Moreover, this is a kind of search that makes use of how your data catalog glossaries are applied. The higher the specificity—that is, the more the terms from the glossaries are actually applied on the assets in the data catalog (using the exhaustivity of the glossaries), the more your recall mechanism will work.

Remember Zipf's law from Chapter 3? If you only rely on technical metadata, your chance of success with block search is low. You need glossary terms applied by humans, not machines, to make your assets distinguishable from each other.

Block search is difficult to build, but it is important to master. In legal, compliance, and complex searches for innovation use cases, block search is the kind of search that will make or break a positive outcome for your company.

And sometimes you have to do a complex search that is not really a well-defined topic with glossary terms assigned to it, but a more haphazard accumulation of things that someone happens to want to know more about.

It is likely that search builders will become even less known and used, because of the evolution of conversational search.

The question is - in our era, could you express a block search through conversational AI? Sure. But chances are, users are not searching like that today. It would look something like in Figure 4-9.

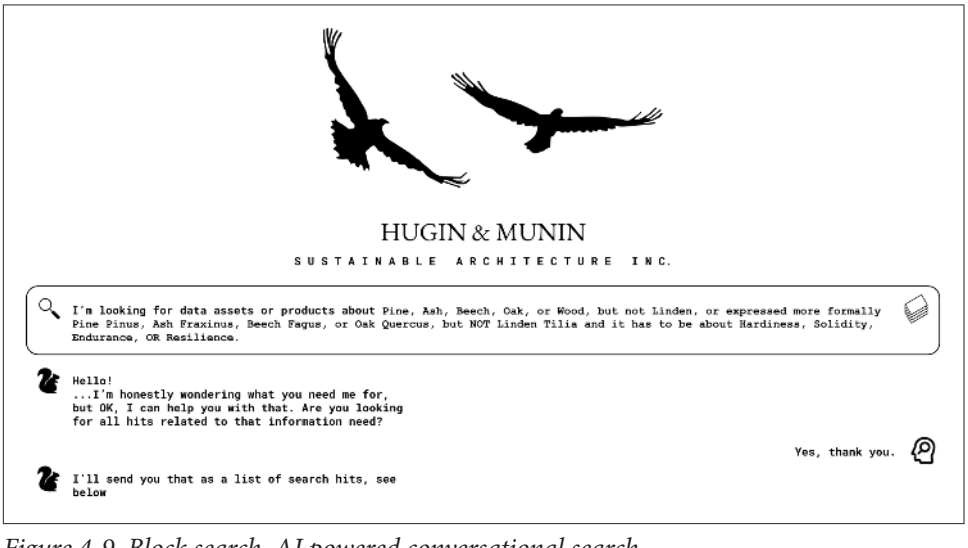


Figure 4-9. Block search, AI powered conversational search

Does this depict reality? Most likely not. At the point in time of writing this book (2020/2026) it is too early to say how you will search for data, conversationally, when translating a structured block search, into conversational search.

From a scientific point of view, it is interesting to see a block search rearchitected into natural language. Why? Because the boolean logic - the ANDs, the ORs, the NOT's - stems from the mathematician George Boole, who established the mathematical expression of logic in language.[1] In a sense, conversational AI is bringing boolean logic closer to its origins - from technical query statements to being discretely built into natural language itself.

Statement Search

Most complex combinatorial searches are *statement searches*: an assorted blend of people, systems, domains, and everything else you can build searches from. These kinds of searches are necessary to make in many disparate situations, ranging from managing the data catalog, to gathering data for a project, to ensuring that assets associated with a given steward who is changing position are passed on to a new steward (for this latter use case, check out Chapter 10 on lifecycles).

Figure 4-10 shows an example of a search performed by the Hugin & Munin data discovery team. They want to find out how many Tableau reports do not have an asset owner in Legal, Finance, or IT.

This search returns all Tableau reports from those departments that have been created after January 1, 2022, that do not have an owner.

The data discovery team will use this search result to reach out to the data stewards for the assets to ask that an owner of the asset be added.

Note that Figure 4-10 is conversational. At this point, you get the logic underneath is doing, you are ready to go!

1 George Boole: *An Investigation of the Laws of Thought* (Cork: Queen's College, 1853), see in particular chap. 4

Figure 4-10. Statement search

Browsing Patterns

Browsing patterns are in fact search patterns, but they usually don't require the end user to phrase search statements (except for glossary lookups). Instead, browsing works by clicking back and forth in either lists of glossary terms, lineage, or graphs. Think of browsing as a phase between other types of search that makes users discover and learn the language and domains of their company. It will allow them to search with more savvy if they can browse the data landscape.

Glossary Browsing

Sometimes, you may just want to explore a topic to better understand a domain. You have many options, but one of the ways is to browse glossaries.

An example from Hugin & Munin is a new employee wanting to better understand how paint is used as the surface treatment for wooden houses. The user types "paint" in the dedicated glossary search bar, as seen in Figure 4-11.

Figure 4-11. Glossary browsing for paint

Here, the difference between the different glossaries stands out clearly: the global glossary is made up of highly controlled terms that apply across the company, the domain glossary refers to a single domain, and the free glossary just adds whatever people like. Clicking deeper into the glossaries reveals the level of formalized structure and organizational reach.

Domain Browsing

Domain browsing is when you go through the capabilities or processes in your company. These kinds of browsings are often driven by lack of context—they allow you to get ideas of where potentially relevant assets could be located. For example, maybe someone is working on a project regarding customer profiles and they want to know if it falls under the purview of Customer Information Management or Customer Preference Management. This might tell them who they need to speak to regarding issues.

They can also just be driven by sheer curiosity—and that kind of browsing is never a waste of time, as it allows you to better understand the data landscape of your company. If you want to see how domain browsing looks, go back to the examples in Chapter 3.

Lineage Browsing

Sometimes, you might want to know where a given asset stems from (upstream), or where it has traveled to (downstream). Browsing upstream in lineage enables you to find out why a given data analytics report is broken. Lineage browsing also allows you to test what the consequences would be downstream of changes in a given asset upstream, if you were to make a change. You could also be browsing lineage to discover potential improvements to existing data processing flows or to discover unused assets in the environment (like a table with flows going in but no flows going out). Or you can search for lineage that has changed (or not changed) in time spans to identify old data pipelines.

A DPO can also document how sensitive data is processed downstream. I show such examples of applied lineage search in Chapter 6.

Remember that lineage functionality will vary from vendor to vendor and that, accordingly, your applied search possibilities will vary: remember to assess lineage functionality in your vendor selection, if this criterion is important to you. This assessment is complex, and it requires substantial time to find the ups and downs of a given lineage functionality.

Graph Browsing

The ultimate way of browsing your data is by visually exploring your knowledge graph—if your data catalog is built on a knowledge graph, as discussed in Chapters 1 and 2. The knowledge graph links all parts of your data catalog beautifully together. It's the manifestation of all the actual nodes in your metamodel. It's the ideal way to maximize serendipity in your search, as you can click your way around everything in the catalog and discover new connections.

Graphs are excellent at providing overviews of social networks. Graphs are used as such in these two sectors, for example:

- Law enforcement, military, and intelligence services
- Universities and academia in general

For *police, military, and intelligence services*, networks of people and the things they use and have (such as phones, weapons, documents) visually laid out in a graph is an absolute must. In police investigations, graphs can map criminal organizations like Mafia families or gangs and help solve the crimes these organizations commit by displaying how people—and networks of people—are linked. Military strategies and tactics on the battlefield are nowadays powered by graphs; they are part of active warfare to map and defeat the enemy. For intelligence services, graphs generate overviews of

networks of extremists under surveillance, such as political or religious extremists. The graph overview helps intelligence agencies to infiltrate and dissolve these networks before they act. Graph solutions for these kinds of organizations are provided by IBM and Palantir, for example.

For *universities and academia in general,* graphs are used to map and visualize networks of researchers or research topics. These are bibliometric maps (sometimes also called *clusters* and *networks*). A beautiful example is this bibliometric cluster of mental health research. Bibliometric maps are used to evaluate the performance of research activities in universities, and also in industrial predictions, since patent clusters indicate what kind of products specific industries are planning to launch.

Knowledge graphs with metadata - like the ones you find in a data catalog - are excellent sources to enable AI use cases. We will discuss this further in part II of the book.

Let's look at an example. In Figure 4-12, a PR manager in the communications department in Hugin & Munin searches for promotion data; some of the search results seem skewed, but it is hard to tell why. The PR manager then searches for "promotion" one more time. The top hit is a dataset with promotion planning details, and the PR manager opens that hit as a graph. The graph visualizes all the terms, processes, and data sources that relate to *promotion*. Suddenly, the PR manager understands why the results are skewed. Someone has added promotion as a free glossary term not to depict communication but to depict career advancement. That term is followed by an exclamation mark (!) because the data catalog automatically detects that it is a duplicate to the domain glossary term, which defines PR activities. Therefore, assets tagged with the free glossary term should be filtered out of the search. With this knowledge, the PR manager can better shape the search to reflect what they are searching for.

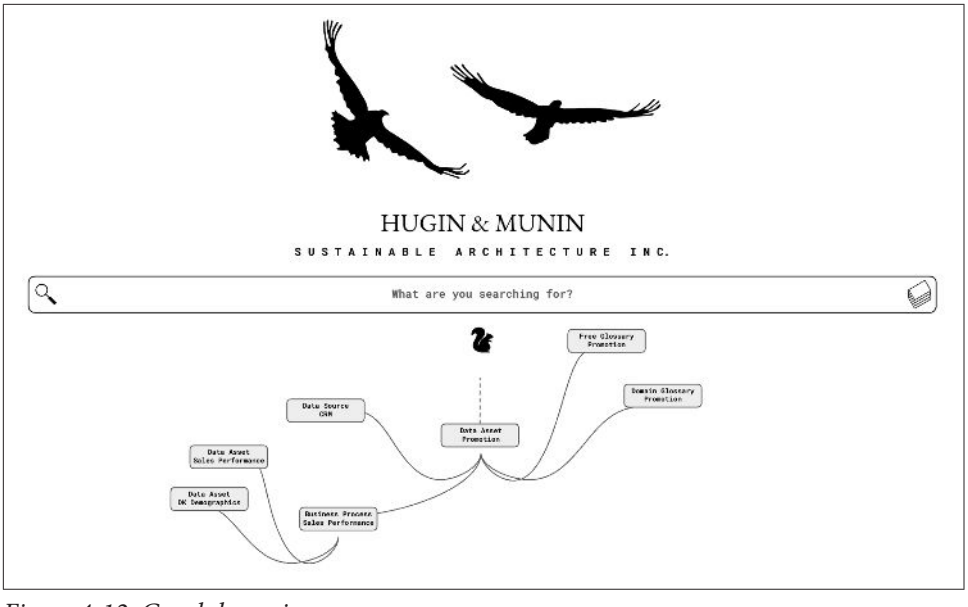


Figure 4-12. Graph browsing

As you have just read, search is a process. In many cases, it will be a series of searches that increase, decrease, and refocus search results, until the search finally matches the information needed.

Searching a Graph-Based Data Catalog

As depicted in Figure 3-3 and 3-4, search can be divided into a spectrum. It goes from easy executable simple searches to more-complex advanced searches. In the latter case, the end user has to remember both the syntax of the IRQL and assess if the semantics of the query statement actually reflect what is being searched for. This is demanding, but useful. The IRQL that the user searches with has been designed by the data catalog provider. It is likely that the IRQL will expand over time, as the technology evolves with the feedback from customers. But an IRQL will never allow you to search for *everything*.

However, for knowledge graph–based data catalogs, it is possible to push search even further and actually search for everything in the data catalog. It requires search skills beyond the IRQL of the specific catalog: instead, here you would have to master the DQL that matches the technology of the catalog in question, for example, GraphQL, SPARQL, Cypher, or Gremlin. Take into consideration that data lineage can be graph-based as well and that, if so, this makes data lineage searchable in a similar fashion.

Searching with a DQL inside a data catalog requires a technical skill set that not all data catalog use cases rely upon. But if you truly want to organize your data just as you like, and search it however you want, then this is what it takes. Think of it like this: an IRQL is always designed by the provider; it will contain some of the elements that are useful to search for, and leave others out. But the graph DQL lets you search for everything you want because it is set up to search for everything that the meta-model contains, however it has been defined.

Summary

This chapter has provided you with examples of patterns for searching for data, in a data catalog. That said, this chapter's biggest takeaway is not the detailed list of those examples. It is this:

- We are in a remarkable situation in the history of search: When it comes to conversational AI, no single vendor controls the market, right now. We are in a situation like the one for search engines, before Google won - also for data catalogs. Things are very uncertain. Nobody really has the definitive answer. Before we know it, that rare moment in time is gone. Let's enjoy it while it lasts!
- That said, conversational search is changing all the patterns of keyword search, so that it becomes more intuitive for the end user to search for data
- The examples in this chapter will move more and more towards conversational search, and the keyword examples will seem increasingly dated.
- Dear reader - please keep experimenting when you search conversationally. Now is the time to do it, we are shaping the future!

The following list of search patterns can both be performed with keywords and increasingly also as conversational search:

- *Basic simple search* is the way of searching that most end users will apply. A well-structured data catalog will deliver precise simple search, especially if it's based on a knowledge graph. But expect a lot of mess deeper down in the search results also.
- *Detailed simple search* requires you to know the syntax of the IRQL in your data catalog. So it takes a little time to write, or just experience, but you get super-precise hits in return.
- *Flexible simple search* also depends on understanding IRQL, but it opens up the search to give more results, increasing your recall and decreasing your precision, while at the same time still being a better way to target a well-defined topic than *basic simple search*.

- *Range search* is searching in intervals, e.g., a time span. This kind of search will result in high precision and low recall.
- *Block search* is a structured way to search for a complex topic using IRQL. It works best if your glossaries are exhaustive and used with great specificity.
- *Statement search* is a way to search for a complex topic; it simply puts a lot of things together in a search. It's not unstructured, but it's haphazard.

You can also visually browse your data catalog:

- *Glossary browsing* is searching in which you go exploring to get informed and enlightened about business terminology.
- *Domain browsing*, *lineage browsing*, and *graph browsing* are ways of searching vertically, horizontally, and relationally, respectively, by clicking through the data landscape.

In the next chapter, we will look at how to observe data, and provide access to it.

About the Author

Ole Olesen-Bagneux is a leading expert in metadata and data cataloging, with over a decade of hands-on experience in data management, governance, and architecture across complex enterprise environments. He holds a PhD in Information Science from the University of Copenhagen, where he also taught courses in Knowledge Organization and Information Retrievalâcore disciplines behind effective data catalogs. As Chief Evangelist at Actian and author of the bestselling first edition of The Enterprise Data Catalog, Ole combines deep academic insight with real-world implementation experience. His work bridges the fields of data engineering, AI, and information science, making him uniquely qualified to redefine how organizations leverage metadata for AI-driven innovation.