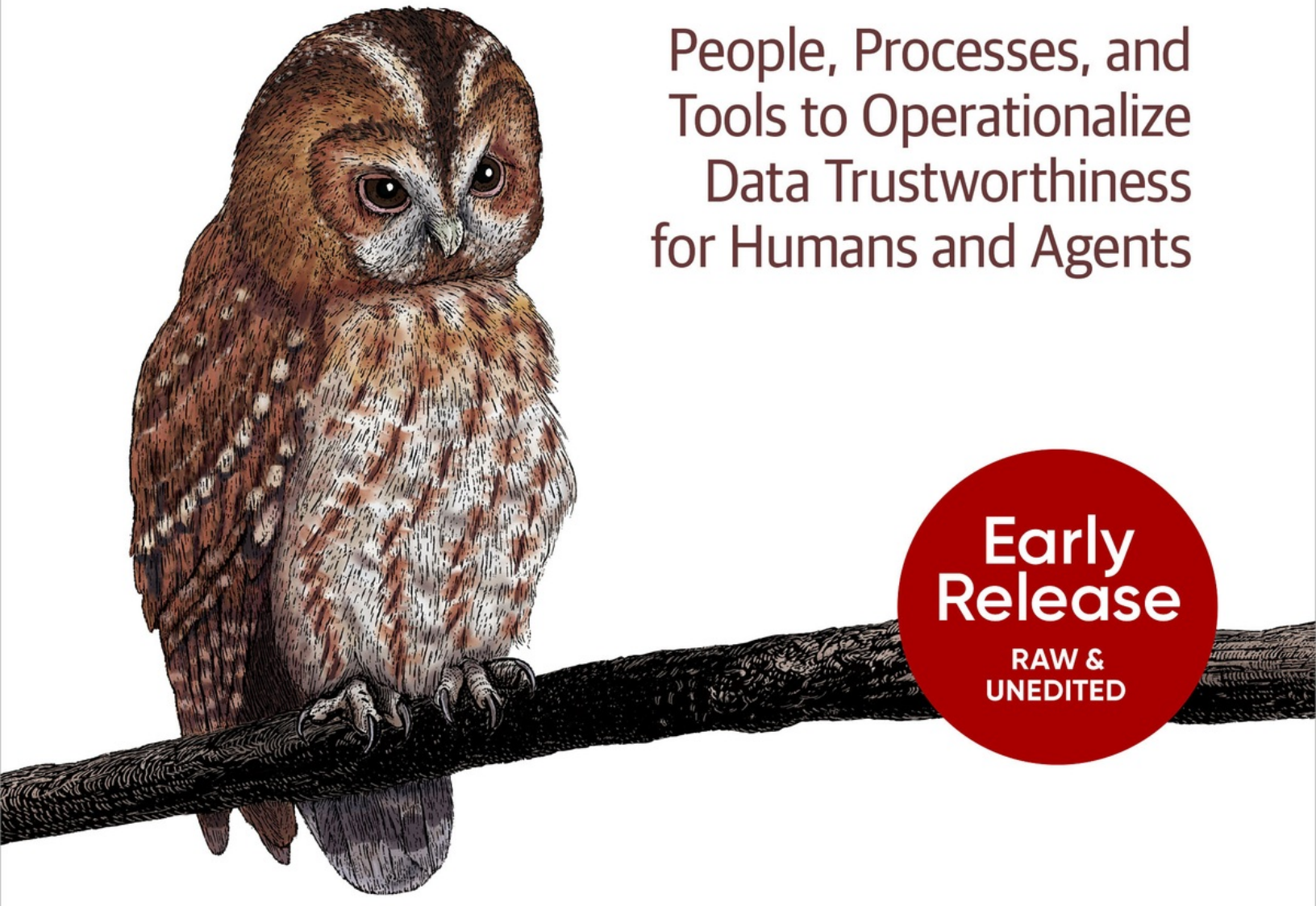
O'REILLY®

2nd Edition

Data Governance in the Era of AI

People, Processes, and Tools to Operationalize Data Trustworthiness for Humans and Agents



Early Release

RAW & UNEDITED

Evren Eryurek, Uri Gilad, Valliappa Lakshmanan,
Dmitry Goodman & Jessi Ashdown

Data Governance in the Era of AI

2ND EDITION

People, Processes, and Tools to Operationalize Data Trustworthiness for Humans and Agents

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

Evren Eryurek, Uri Gilad, Valliappa Lakshmanan, Dmitry Goodman, and Jessi Ashdown

Data Governance in the Era of AI

by Evren Eryurek, Uri Gilad, Valliappa Lakshmanan, Dmitry Goodman, and Jessi Ashdown

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: Steve Hayes

Development Editor: Corbin Collins

Production Editor: Destiny Baitinger

Interior Designer: David Futato

Interior Illustrator: Kate Dullea

March 2021: First Edition

May 2027: Second Edition

Revision History for the Early Release

- 2026-09-08: First Release

See https://oreilly.com/catalog/errata.csp?isbn=9798295901201 for release details.

979-8-295-90116-4

Brief Table of Contents (*Not Yet Final*)

Chapter 1. Ingredients of Data Governance: Tools

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. Please note that the GitHub repo will be made active later on.

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

A lot of the tasks related to data governance can benefit from automation. Machine learning (ML) tools and automatic policy applications or suggestions can accelerate data governance tasks. In this chapter we will review some of the tools commonly referred to when discussing data governance.

When evaluating a data governance process/system, pay attention to the capabilities mentioned in this chapter. The following discussion concerns tasks and tools that can augment complete end-to-end support for the processes involved in, and the personnel responsible for, data governance organization. We'll dive into the various processes and solutions in more detail in later chapters.

Since this book first appeared, one shift has reshaped almost every tool in this chapter: machine learning and generative AI have become first-class consumers—and producers—of enterprise data. AI systems feed on unstructured data (documents, images, audio, and video) and on a new,

derived artifact, the *vector embedding*, that earlier governance tooling never had to account for. AI has also become both a tool and an object of governance: the same large language models (LLMs) that now work to classify and catalog data are themselves systems whose training data, prompts, and outputs must be governed. Furthermore, the entity requesting data is, more and more often, not a person but a service or an autonomous agent acting on someone's behalf. We weave these themes through the sections that follow rather than isolating them, because they touch nearly every ingredient of data governance.

The Enterprise Dictionary

To begin, it is important to understand how an organization works with data and enables data governance. Usually, there is an *enterprise dictionary* or an enterprise *policy book* of some kind.

The first of these documents, the enterprise dictionary, is one that can take many shapes, from a paper document to a tool that encodes or automates certain policies. It can also be partial, or absent, or duplicated across business units, all of which are common and none of which put data governance out of reach. We return to those cases at the end of this section. The Enterprise Dictionary is an agreed-upon repository of the information types (*infotypes*) used by the organization—that is, data elements that the organization processes and derives insights from. An infotype will be a piece of information with a singular meaning—"email address" or "street address," for example, or even "salary amount."

In order to refer to individual fields of information and drive a governance policy accordingly, you need to name those pieces of information. An organization's enterprise dictionary is normally owned by either the legal department (whose focus would be compliance) or the data office (whose focus would be standardization of the data elements used).

In Figure 1-1, you can see examples of infotypes, and a potential organization of those infotypes into organization-specific *data classes* to be used in policy making.

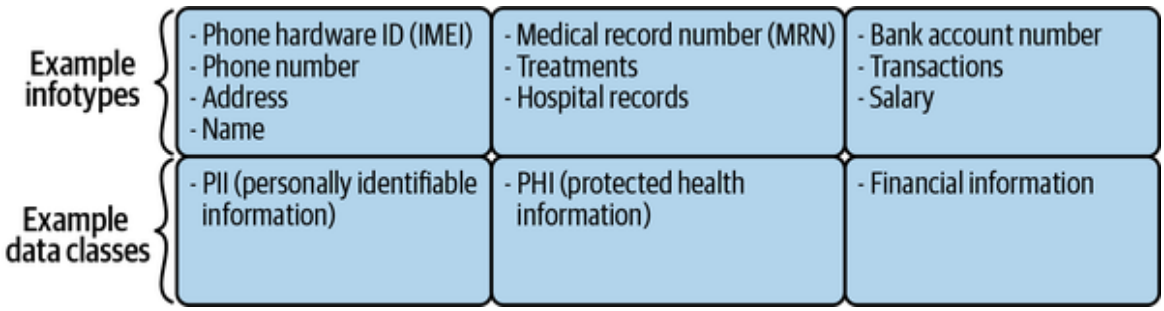


Figure 1-1. Data classes and infotypes

Once the enterprise dictionary is defined, the various individual infotypes within it can be grouped into data classes, and a policy can be defined for each data class. The enterprise dictionary generally contains data classes, policies related to the data classes, and additional metadata. The following sections expand on this.

Data Classes

A good enterprise dictionary will contain a listing of the classes of data the organization processes. Those will be infotypes collected into groups that are treated in a common way from the policy management aspect. For example, an organization will not want to treat "street addresses," "phone numbers," "city, state," and "zip code" differently in a granular manner but must rather be able to set a policy that "all location information for consumers must be accessible only to a privileged group of personnel and be kept only for a maximum of 30 days." This means that the enterprise dictionary we've just described will actually contain a hierarchy of infotypes—at the *leaf nodes* there will be the individual infotypes (e.g., "address," "email"), and at the *root nodes* you will find a data class, or a sensitivity classification (or sometimes both).

Figure 1-2 shows an example of such a hierarchy from a fictional organization.

Figure 1-2. A data class hierarchy

In the data class hierarchy detailed in Figure 1-2, you can see how infotypes such as IMEI (cellular device hardware ID), phone number, and IP address are grouped together under personally identifiable information (PII). For this organization, these are easily identifiable automatically, and policies are defined on "all PII data elements." PII is paired with PHI (protected health information) in the "restricted data" category. It is likely that there are further policies defined on all data grouped under the "restricted" heading.

Data classes are usually maintained by a central body within the organization, because policies on "types of data classes" usually affect compliance to regulation.

Here are some example data classes seen across many organizations:

PII

PII stands for “Personally Identifiable Information”. PII is data such as name, address, and personal phone number that can be used to uniquely identify a person. For a retailer, this can be a customer list. Other examples include lists of employee data, a list of third-party vendors, and similar information. There are many regulations around the world that place restrictions on the sharing of PII, in addition, different jurisdictions (EU, US, India, etc.) have different laws governing its sharing and use.

Financial information

This is data such as transactions, salaries, benefits, or any kind of data that can include information of financial value. This is of utmost importance to businesses that need to comply with minimum data retention policies and other fraud-related laws.

Business intellectual property

This is information related to the success and differentiation of the business.

The variety and kind of data classes will change with the business vertical and interest. The preceding example data classes (PII, financial, business intellectual property) will not work for every organization, and the meaning of data collected and the classification will differ between businesses and between countries. Do note that data classes are a combination of information elements belonging to one topic. For example, a phone number is usually not a data class, but PII (of which phone number is a member) is normally a data class.

The characteristics of a data class are twofold:

- A data class references a set of *policies*: the same retention rules and access rules are required on this data.

- A data class references a set of individual infotypes, as in the example in Figure 1-1.

WHAT IF I DON'T HAVE AN ENTERPRISE DICTIONARY?

Not everybody's situation is so organized as the state we defined so far. In fact, many organizations' reaction to the previous section is "our data is nothing like that". You are in good company. The Enterprise dictionary is a destination, not a prerequisite. Very few of the organizations we encountered had a dictionary on day one, and there are a few examples where they still do not have a single source of truth. There are three common situations apart from the ideal scenario we have described.

The first situation is having no dictionary. The way forward in this case is not(!) to set up a "dictionary committee" and spend years naming every single data class and infotype. Rather, prioritize: pick the highest-risk data classes, which will probably be PII and whatever the regulator asks about first, and define the infotypes related to that data class. Leave the rest of the data unnamed for now. A dictionary with only a few entries, but entries people actually use, is much better than a perpetual draft.

The second case is scale. If you are ingesting petabytes of events every hour, and you simply cannot classify every record on the way in, and evaluation of each individual row causes unacceptable latency, you have a scale problem. The move here is to push the policy decision and evaluation from the individual record and into the container: classification of a stream sample rather than an entire stream, and attaching the policy to the table, topic or bucket scales better. Reserve the per-record treatment for the slice of data that is high risk and thus warrants it. Also - you can batch classification tasks to an offline/overnight process - the cases where you need an answer to a retention or access question in milliseconds are relatively rare - that analyst who opened a ticket for table access will be serviced in due course.

The third case - and the messiest one - is having multiple enterprise dictionaries. This is common when a company grows through

acquisitions but even if data governance is not centrally managed but is per-division. We have seen organizations carry three or four or more "unified" data models, none of which are actually unified. Merging disparate enterprise dictionaries is a project that tends to never end, and does not have a clearly understood benefit to the end-users - meaning it will not necessarily be prioritized. The practical move here is to leave the local dictionaries where they are, and build a bridge, or a mapping, between them. For example - connect `customer_email` in one dictionary to `EmailAddr` in the other dictionary so that they resolve to the same infotype. Policy will then attach to that shared infotype, and each business unit keeps the vocabulary it is already using.

The tools in the rest of this chapter assume some agreed-upon vocabulary. We will never assume the work is done and settled.

Data Classes and Policies

Once the data the organization handles is defined in an enterprise dictionary, policies that govern the data classes can be assigned to data across multiple containers. The desired relationship is between a policy (e.g., *access control*, *retention*) and a data class rather than a policy and an individual container. Associating policies with the meaning of data allows for better human understanding ("Junior analysts cannot access PII" versus "Junior analysts cannot access column #3 on Table B") and supports scaling to larger amounts of data.

So far, we have discussed the relationship between data classes and policies. Frequently, along with the data class specification, the central data office, or legal, will define an enterprise *policy book*. An organization needs to be able to answer the question, "What kinds of data do we process?" An enterprise policy book records that. It specifies the data classes the organization uses, the kinds of data that are processed, and how they are processed, and it elaborates on "what are we allowed and not allowed to do" with the data. This is a crucial element in the following respects:

- For compliance, the organization needs to be able to prove to a regulator that it has the right policies in place around the handling of data.
- A regulator will require the organization to submit the policy book, as well as proof (usually from audit logs) of compliance with the policies.
- The regulator will require evidence of procedures to ensure that the policy book is enforced and may even comment on the policies themselves.

TIP

We'd like to stress the importance of having a well-thought-out and well-documented enterprise policy book. Not only is it incredibly helpful for understanding, organizing, and enforcing your policies, but it also enables you to quickly and effectively react to changing requirements and regulations. Additionally, it should be noted that having the ability to quickly and easily provide documentation and proof of compliance—not only for external audits but also for internal ones—should not be discounted. Knowing at a glance how your company is doing with its policies and its management of those policies is critical for ensuring a successful and comprehensive governance program. During the course of our research and interviews, many companies expressed their struggles with being able to conduct quick internal audits to know how their governance strategy was going. This often resulted in much time and effort being spent backtracking and documenting policies, how they were enforced, on what data they were attached, and so on. Creating an enterprise policy book and setting yourself up to more easily audit internally (and thus externally) will help you avoid this pain.

To limit liability, risk management, and exposure to legal action, an organization will usually define a maximum (and a minimum) retention rate for data. This is important because during an investigation, certain law enforcement agencies will require certain kinds of data, which the organization must therefore be able to supply. In the case of financial institutions, for example, it is common to find requirements for holding certain kinds of data (e.g., transactions) for a minimum of seven years. Other kinds of data pose a liability: you cannot leak or lose control of data that you don't have.

Data-combination policies are another kind. Such a policy might assign a low sensitivity to first name and last name separately, but a much higher one to the combination, which is far more identifying.

Another kind of policy will be *access control*. For data, access control goes beyond "yes/no" and into *no access*, *partial access*, or *full access*. Partial access is accessing the data when some bits have been "starred out," or accessing the data after a deterministic encryption transformation, which will still allow acting on distinct values, or grouping by these values, without being exposed to the underlying cleartext. You can think of partial access as a spectrum of access, ranging from zero access to ever-increasing details about the data in question (format only, number of digits only, tokenized rendition, to full access). See Figure 1-3.

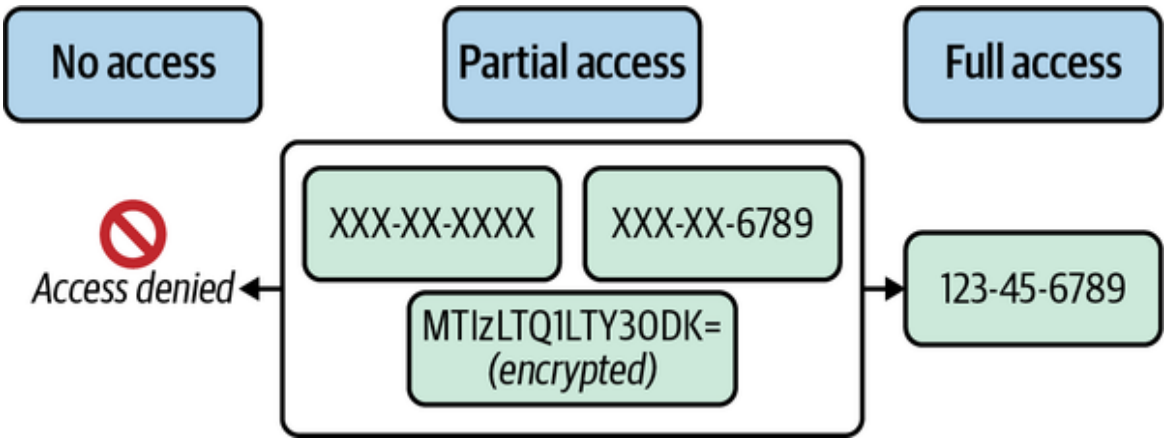


Figure 1-3. Examples of varying levels of access for sensitive data

Normally, a policy book will specify the following:

- Who (inside or outside the organization) can access a data class
- The retention policy for the data class (how long data is preserved)
- Data residency/locality rules, if applicable, and which system can process what data
- How the data can be processed (OK/Not OK for analytics, machine learning, etc.)
- Other considerations by the organization

So far, we have introduced several terms in quick succession, lets pin down how these relate before we move on. The Enterprise Dictionary, the Policy Book and the Data Catalog are not interchangeable and they do not nest inside each other.

The Enterprise Dictionary is the vocabulary. It defines which infotypes exist and which data classes these infotypes a data class consists of. The main value of an Enterprise dictionary is meaning - what kind of data is actually in a data class. The dictionary does not say anything about the data you actually have across the enterprise.

The policy book is the rulebook. It attaches rules (access control, retention, use case, data residence) to the data classes in the dictionary. Usually, the dictionary and policy book are owned by the same central body, which is why they are sometimes referred to as one artifact.

The data catalog is the inventory. It is the only one of the three that knows about real tables, files, streams, and vector stores. It keeps a record of each asset's location, ownership, and data classes contained within the asset. The association between asset and data class is how a policy is enforced on a specific column in a specific table.

To put it plainly: the dictionary defines the words, the policy book writes the rules for using these words, and the catalog is where the rules meet the actual data. Changing a rule in the policy book triggers an operation across all relevant assets in the data catalog without any edits to a particular table (assuming everything is working correctly)

Now let's discuss specific tools and functionality that can accelerate data governance work and optimize personnel time.

Per-Use-Case Data Policies

Data can have different meanings and require different policies when the data use case is taken into consideration. An illustrative example might be a furniture manufacturer that collects personal data (names, addresses, contact numbers) in order to ensure product delivery. The very same data

can potentially be used for marketing purposes, but consent, in most cases, is not guaranteed for the purpose of marketing at the time a purchase is made. However, at the same time, you would very much like that sofa to be delivered to your home, so you do want the manufacturer to store your address! The use case, or purpose, of the data access ideally should be an overlay on top of your organizational membership and organizational roles. One way to think about this would be as a "window" through which the analyst can select data, specifying the purpose ahead of time, and potentially moving the data into a different container for that purpose (the marketing database, for example), all with an audit artifact and lineage tracking that will be used for tracking purposes.

A more recent example comes from AI. Suppose an employee is permitted to open and read a particular customer contract in a document management system. It is tempting to assume that because the document is "accessible" to them, it is fair game for any downstream use—but the purpose still matters. Using that contract to answer a question in a retrieval-augmented generation (RAG) assistant is a markedly different purpose from using it as training data for a foundation model, which is different again from exporting it to a vendor. The organization may have every right to let a person read the document and still lack the legal or contractual basis or the data subject's consent to feed it into model training, where its contents can be memorized and unexpectedly resurface in a model's outputs. The purpose of "view for contract negotiation" is not the same as "use the data for training" even though in both cases that data is only read by a person, or a system. A purpose-aware policy lets you grant the first while withholding the second.

This example also surfaces a concept that recurs later in the chapter: the request to open the contract may come not from the employee directly but from an AI assistant or agent acting on their behalf. We call this on-behalf-of (OBO) access. The governance question becomes: when a service performs an action, can we attribute that action—and its purpose—back to the human principal who triggered it? Purpose and identity are now

entangled, and we return to the identity side of this problem in "IAM—Identity and Access Management" later in this chapter.

IMPORTANCE OF USE CASE AND POLICY MANAGEMENT

Increasingly, as requirements and regulations change to accommodate the new and different types of data that are collected, the use case of data becomes an important aspect of policy management. One of the things that we've heard companies say over and over is that the use case of data needs to be a part of policies—the simple "data class to role-based access type" isn't sufficient. We've given the example of a furniture manufacturer and access to, and usage of, customer address for delivery versus marketing purposes. Another example we've heard about throughout our research is that of a company that has employees who may perform multiple roles. In this case, a simple role-based access policy is inadequate, because these employees may be allowed access to a particular set of data for performing tasks relating to one role but may not have access for tasks relating to a different role. From this you can see how it's more efficient (and effective) to consider access related to what the data will be used for—its use case—rather than only considering infotype/data class and employee role.

Data Classification and Organization

To control the governance of data, it is beneficial to automate, at least in part, the classification of data into, at the very least, infotypes—although an even greater automation is sometimes adopted. A data classifier will look at unstructured data, or even a set of columns in structured data, and infer "what" the data is—e.g., it will identify various representations of phone numbers, bank accounts, addresses, location indicators, and more.

Cloud providers offer data-loss-prevention and classification services such as Google's Cloud Data Loss Prevention (DLP) (now part of Sensitive Data

Protection). Another classifier is Amazon's Macie.

Until recently, classifiers like these worked largely by pattern matching: regular expressions and dictionaries that recognize the shape of a Social Security number, the checksum of a credit card, or the format of an email address. That approach is fast and cheap, but it is blind to meaning. It cannot tell that a paragraph describes a planned layoff, that a support ticket is a veiled legal threat, or that a sentence is sarcastic rather than sincere—distinctions that increasingly matter for governance.

LLMs have begun to fill that gap. Because an LLM reads context rather than just format, AI is now one of the most capable classification tools available. An LLM can classify highly nuanced categories—"strategic business intent," "contains a contractual commitment," "negative sentiment expressed sarcastically"—that no regular expression could capture. The major cloud providers' sensitive-data-protection and cataloging services, along with dedicated data-governance platforms, now layer machine-learning and LLM-based classification on top of traditional pattern matching.

This capability comes with two cautions. First, an LLM classifier is more expensive, slower, and less deterministic than a regular expression; you will often want a tiered approach—cheap pattern matching first, the model reserved for the ambiguous cases. Second, and this is the recurring theme of AI as both tool and object: the classifier itself reads every record it inspects. A model that consumes your most sensitive data in order to label it is now part of your attack surface and your compliance scope. Where that model runs, what it logs, and whether your data leaves your boundary to reach it are themselves governance decisions.

Automation of *data classification* can be accomplished in two main ways:

- Identify data classes on ingest, triggering a classification job on the addition of data sources
- Trigger a data-classification job periodically, reviewing samples of your data

When it is possible, identifying new data sources and classifying them as they are added to the data warehouse is most efficient, but sometimes this is not possible with legacy or federated data.

Upon classifying data, you can do the following, depending on the desired level of automation:

Tag the data as “belonging to a class” (see The Enterprise Dictionary)

Automatically (or manually) apply policies that control access to and retention of the data according to the definition of the data class, “purpose,” or context for which the data is accessed or manipulated

Unstructured Data, Embeddings, and Open Table Formats

Most of the tooling described so far grew up around structured data: tables, columns, schemas—but the center of gravity has shifted. The data that fuels modern AI is overwhelmingly unstructured: documents, images, audio, and video now make up the majority of most organizations’ information, and they are the input that LLMs consume directly. Governing it well means accounting not only for these raw assets but for two artifacts the AI era has made first-class: the vector embedding derived from content, and the open table formats increasingly used to store data at scale. Each is recent enough that many catalogs and policy engines still cannot see it, and each, as we will show, inherits the governance obligations of its sources. We introduce all three here in this section, before the catalog discussion that follows, because everything downstream—cataloging, quality, lineage, and access—now has to reach them.

Vector Embeddings and Vector Databases

An *embedding* is a numerical representation of a piece of content (a paragraph, an image, a recording) that is produced by a model so that semantically similar items sit close together in a high-dimensional space. Embeddings are the backbone of semantic search and RAG, and they live in

vector databases: purpose-built systems, or vector extensions added to familiar relational and search engines. From a governance standpoint, a vector store is just another container of data—but its metadata needs are different. For each collection of embeddings you want to record the usual ownership and sensitivity, but also which model produced the vectors, how the source content was chunked, and—most importantly—a link back to the source documents, so the lineage from a governed record to its vector representation is never lost. Figure 1-4 shows the embedding pipeline.

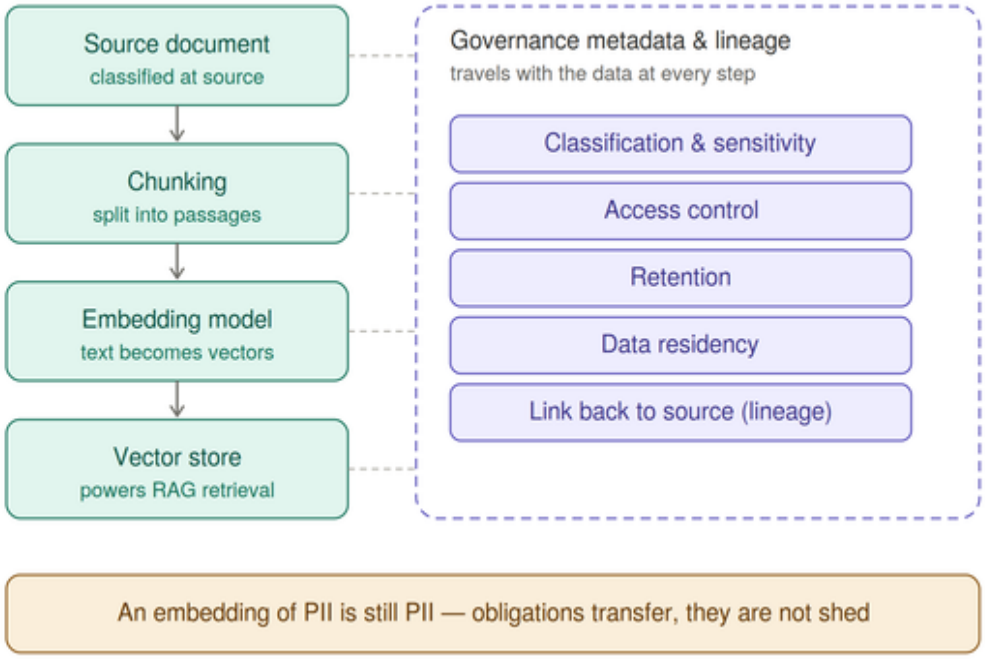


Figure 1-4. Governance metadata and lineage travel with data through the embedding pipeline

Open Table Formats as Governance Infrastructure

One more development belongs here, because it is often filed under "storage optimization" when it is really governance infrastructure. Open table formats such as Apache Iceberg, Delta Lake, and Apache Hudi sit on top of inexpensive object storage and add a transactional metadata layer that turns a pile of files into a properly governed table. Their feature lists read like a catalog of governance primitives: schema evolution (changing a

table's shape without breaking consumers' workflows), time travel (querying the table as it existed at a past point in time), partition pruning, and ACID transactions at scale, with the table's metadata co-located with its data.

For governance, time travel is the most consequential of these. Querying historical snapshots of a table—and seeing how its schema evolved—supports exactly the point-in-time auditing and reproducibility that regulators ask for, and it dovetails with the temporal dimension of lineage discussed later in this chapter. Because these formats are open and engine-agnostic, they also lend themselves to multi-cloud deployment: the same table can be read by multiple engines across providers, which means governance controls can be defined against the table itself rather than re-implemented for each proprietary warehouse. A growing layer of open catalog services manages these tables and centralizes access policy across engines.

Do Policies Survive Transformation?

Across all of these artifacts, one question recurs and is easy to get dangerously wrong: when data is transformed into a new representation, do the policies that governed the original still apply? The temptation to answer "no" is sharpest with embeddings. An embedding looks anonymized—it is, after all, just a vector of numbers with no obvious name or address in it. That intuition is wrong. Embeddings are not one-way hashes. A class of techniques known as embedding-inversion attacks can reconstruct the original text from its vector with surprising fidelity; in one published experiment, roughly 40% of the sensitive items in sentence-length text were recovered exactly, using a readily available open-source model. The OWASP Top 10 for LLM Applications now lists vector and embedding weaknesses as a distinct risk category for exactly this reason. See Tonic.ai, Sensitive Data in Text Embeddings Is Recoverable.

The conclusion is straightforward, and it generalizes: an embedding of PII is still PII, and a vector store built from restricted documents inherits the restrictions of its sources. Classification, access control, retention, and

residency obligations all transfer to the new representation; they are not shed in the transformation. The same logic extends beyond privacy—an embedding of copyrighted text does not launder its copyright, and an embedding of a trade secret is still a trade secret. Treat a vector store with the same rigor as the source it was built from, and make sure your classification and policy tooling can follow data across this transformation. We will return to this principle throughout the rest of the chapter.

Data Cataloging and Metadata Management

When talking about data, data classification, and data classes, we need to discuss the *metadata,* or the "information about information"—specifically, where it's stored and what governance controls there are on it. It would be naive to think that metadata obeys the same policies and controls as the underlying data itself. There are many cases, in fact, where this can be a hindrance. Consider, for example, searching in a metadata catalog for a specific table containing customer names. While you may not have access to the table itself, knowing such a table exists is valuable (you can then request access, you can attempt to review the schema and figure out if this table is relevant, and you can avoid creating another iteration of this information if it already exists).

Another example is *data residency*–sensitive information which must not leave a certain national border. But that restriction does not necessarily apply to information about the existence of the data itself, which may be relevant in a global search. A final example is information about a listing of phone calls (who called whom, from where, and when), which can be potentially more sensitive than the actual calls themselves, because a call list can indicate communications between certain people at certain times, and is an indicator for the potential of an information exchange, even if the information itself is not part of the record.

Crucial to metadata management is a *data catalog*, a tool to manage this metadata. Whereas enterprise data warehouses, such as Google BigQuery, are efficient at processing data, you probably want a tool that spans multiple

storage systems to hold the information about the data. This includes where the data is and what technical information is associated with it (table schema, table name, column name, column description)—but you also should allow for the attachment of additional "business" metadata, such as who in the organization owns the data, whether the data is locally generated or externally purchased, whether it relates to production use cases or testing, and so on.

As your data governance strategy grows, you will want to attach the particulars of data governance information to the data in a data catalog: *data class*, *data quality*, *sensitivity*, and so on. It is useful to have these dimensions of information schematized, so that you can run a faceted search like "Show me all data of type:table and class:X in the 'production' environment."

Everything in this section now has to extend to the artifacts introduced earlier. A modern catalog indexes not only tables and files but the vector stores and open-format tables described in the previous section, applies the same faceted search to them — "show me all collections of class:X derived from production sources" — and carries their governance metadata (source model, chunking, lineage back to origin) alongside the classic schema fields.

The data catalog clearly needs to efficiently index all this information and must be able to present it to the users whose permissions allow it, using high-performing search and discovery tooling.

Data Assessment and Profiling

A key step in most insight generation workflows, as you sift through data, is to review that data for outliers. There are many possible reasons for outliers, and best practice is to review before making sweeping/automated decisions. *Outliers* can be the result of data-entry errors or may just be inconsistent with the rest of the data, but they can also be weak signals or less-represented new segments or patterns. In many cases, you will need to normalize the data for the general case before driving insights. This

normalization (keeping or removing outliers) should be done in the context of the business purpose the data is being used for—for example, if you are looking for unusual patterns or not.

The reason for normalizing data is to ensure data quality and consistency (sometimes data entry errors lead to data inconsistencies). Again, this must be done in the context of the business purpose of the data use case. Data quality cannot be performed without a directing business case, because "reviewing transactions" is not the same for a marketing team (which is looking for big/influential customers) as it is for a fraud analysis team (which is looking for provable indications of fraud).

Note that ML models, for example, are susceptible to arriving at the wrong conclusions by extracting generalizations from erroneous data. This is also true for many other types of analytics.

Data engineers are usually responsible for producing a report that contains data outliers and other suspected quality issues. Unless the data errors are obviously caused by data entry or data processing, the data errors are fixed/cleansed by data analysts who are part of the organization that owns/produces the data, as previously mentioned. This is done in light of the expected use of the data. However, the data engineers can leverage a programmatic approach in fixing the data errors, as per the data owners' requests and requirements. The data engineers will look for empty fields, out-of-bounds values (e.g., people of ages over 200 or under 0), or just plain errors (a string where a number is expected). There are tools to easily review a sample of the data and make the cleanup process easier, for example, Anomalo and Stitch.

These cleanup processes work to ensure that using the data in applications, such as generating a machine learning model, does not result in it being skewed by data outliers. What they do not do is reduce the amount of data under governance, as we note shortly. Ideally, data should be profiled in order to detect anomalies per column and make a determination on whether anomalies are making sense in the relevant context (e.g., customers shopping in a physical store outside of store hours are probably an error,

while late-night online ordering is very much a reality). The bounds for what kinds of data are acceptable for each field are set, and automated rules prepare and clean up any batch of data, or any event stream, for ingestion. Care must be taken to avoid introducing bias into the data, such as by eliminating outliers where they should not be eliminated.

One potential outcome of the cleanup described here is important to keep an eye on: while the assumption that “clean data is easy to govern” is sound, note that the cleanup actually resulted in a new artifact: the clean dataset, while you still hold the raw records. Both datasets are real assets, both datasets share the same sensitivity. Everything we have discussed so far now needs to be applied both to the raw dataset and to the cleaned dataset. Both require access control policies, retention expiration dates, and if a deletion request is issued - it needs to reach both. If a customer asks to be forgotten and you remove them from the curated table while keeping raw records, their request was not fully respected. The new copy and the old raw copy also can potentially disagree with each other - that was the whole point of the cleanup. There is an advantage in preserving this disagreement: if, down the line, someone is reading a dashboard and needs to be able to tell where the data came from, that raw dataset can be useful. Lineage, discussed later in this chapter, is how we arrive at the answer. The same principle is applied to embeddings: a new representation of governed data is still governed data, and the transformation is when you attach the metadata rather than the moment you lose it. None of this is of course an argument against cleaning up the data - cleanup is necessary, sometimes mandatory. This is a note about planning: plan to keep the original raw dataset and decide in advance on the retention period. Be ready to have two copies of the data for as long as that is useful.

For data destined to train ML models, the idea of “quality” has to stretch in a direction the term does not usually imply. A dataset can be free of empty fields, out-of-bounds values, and type errors and still be badly flawed for AI, because it encodes historical bias or contains toxic content. A hiring dataset that faithfully records years of skewed past decisions is “clean” by every traditional measure and will still teach a model to discriminate; a

web-scraped text corpus may be well-formed and still riddled with hate speech. Profiling for AI therefore has to include profiling for representational balance and for harmful content, before the data reaches a training pipeline.

A growing set of tools supports this. Open-source fairness toolkits such as Google's What-If Tool and Microsoft's fairlearn and the bias-measurement features built into the major ML platforms measure how a dataset—and a model trained on it—behaves across demographic groups, surfacing imbalances and disparate outcomes. Content classifiers and toxicity scorers, together with data-centric quality tools that flag mislabeled or harmful examples, handle the content side. The point is that the bias warning we just gave about outlier removal generalizes: for AI training data, quality assessment and fairness assessment are the same activity, and both have to happen before, not after, the model learns from the data.

Data Quality

Data quality is an important parameter in determining the relevant use cases for a data source, as is being able to rely on data for further calculations/inclusions with other datasets. You can identify data quality by looking at the data source—i.e., understanding where it physically came from. (Error-prone human entry? Fuzzy IoT devices optimizing for quantity, not quality? Highly exact mobile app event stream?) Knowing the quality of data sources should guide the decision of whether to join datasets of varying quality, because low-quality data will reduce confidence in higher-quality sources. Data quality management processes include creating controls for validation, enabling quality monitoring and reporting, supporting the triage process for assessing the level of incident severity, enabling root cause analysis and recommendation of remedies to data issues, and data incident tracking.

There should be different confidence levels assigned to different quality datasets. There should also be considerations around allowing (or at least curating) resultant datasets with mixed-quality ancestors. The right

processes for data quality management will provide measurably trustworthy data for analysis.

One possible process that can be implemented to improve data quality is a sense of ownership: making sure the business unit responsible for generating the data also owns the quality of that data and does not leave it behind for users downstream. The organization can create a data acceptance process wherein data is not allowed to be used until the owners of the data prove the data is of a quality that passes the organization's quality standards.

Data Poisoning and Adversarial Injection

Everything so far treats data-quality problems as accidents: human entry errors, fuzzy sensors, inconsistent formats. AI introduces a category that is no accident: *data poisoning*. When an organization's analytics or models learn from data it does not fully control—public web pages, user-generated content, third-party feeds—adversaries have an incentive to corrupt that data deliberately, so the resulting model behaves the way they want. Quality, for AI, becomes a security property as much as an accuracy one.

The manipulation is already commercial and routine. In 2026, moderators of a large biohacking community on Reddit (*r/Biohackers*) began restricting certain posts after discovering that peptide and hormone-therapy companies were seeding the forum with promotional content—not to persuade human readers, but to influence what AI assistants would later say about their products. The practice even has a name in marketing circles—*answer-engine optimization* (AEO)—and firms openly sell it. 404 Media documented RedRover, which advertises an 'army of agents' publishing blog and Reddit posts to do SEO and AEO at scale, pitching simultaneous placement in Google and ChatGPT answers. Even mainstream vendors have entered the space—HubSpot now sells a dedicated AEO product.

Data poisoning works because AI search systems lean heavily on user-generated sites: researchers at Cornell (Zhang, Triedman, and Shmatikov) have shown that a snippet as short as 13 words, planted on a site like Reddit

or Wikipedia, can steer an AI system's answers across a whole cluster of related questions. A handful of poisoned sentences can shape outputs at scale. See the article "Deep-Research Agents Can Be Poisoned via User-Generated Content," by Tingwei Zhang, Harold (Hal) Triedman, and Vitaly Shmatikov.

The governance response borrows from data quality and extends it. Incoming data streams—especially those feeding retrieval systems or training pipelines—should be profiled not only for the familiar quality issues but for signs of coordinated manipulation: sudden bursts of near-duplicate content, anomalous provenance, text that appears written to be scraped rather than to be read. Provenance and source reputation become quality signals in their own right. And as with bias, prevention at ingestion is far cheaper than undoing the damage after a model has already absorbed it—which takes us directly into lineage.

Lineage Tracking

Data does not live in a vacuum; it is generated by certain sources, undergoes various transformations, aggregates additional data, and is eventually supporting certain insights. A lot of valuable context is generated from the source of the data and how it was manipulated along the way, which is crucial to track. This is *data lineage*.

Why is lineage tracking important? One reason is understanding the quality of a resulting dashboard/aggregate. If that end product was generated from high-quality data, but later the information is merged into lower-quality data, that leads to a different interpretation of the dashboard. Another example is viewing, in a holistic manner, the movement of a sensitive data class across the organization data scape, to make sure sensitive data is not inadvertently exposed into unauthorized containers.

Lineage tracking should be able, first and foremost, to present a calculation on the resultant metrics, such as quality, or on whether or not the data was tainted with sensitive information. And later it must be able to present a

graphical graph of the data traversal itself. This graph is very useful for debugging purposes but is less so for other purposes.

LINEAGE TRACKING AND TIME/COST

When describing why lineage tracking, especially lineage visualization, is so important, companies have often referred to the level of effort they have to put into debugging and troubleshooting. They have stated how much time is spent not on fixing problems or errors but just on trying to find out if there are any errors or problems at all. We have heard time and time again that better tracking (e.g., notifications about what's going on and when things are on fire and should be looked at) not only would help solve issues better but also would save valuable time , and expense, in the long run. Often when lineage is talked about, the focus is on knowing where data came from and where it's going to, but there is also value in visually seeing/knowing when and where something breaks and being able to take immediate action on it.

Lineage tracking has applications well beyond data governance, and in practice it is delivered by two different techniques that are easy to conflate. Tracing movement answers "where did this come from and where did it go." Snapshotting answers a different question: "what did this look like at the moment the decision was made." Most of the use cases that follow need both.

Key use case	Objective	Scenarios	Common implementation techniques
Root cause analysis, debugging	Identify steps/locations where a data failure occurred.	**Broken Dashboards / Pipelines:** Trace corrupt metrics back to faulty upstream ETL steps. **Data Drift & Anomalies:** Identify unexpected distribution shifts across pipelines.	**Visualization:** Interactive upstream DAG exploration with status overlay (red/green nodes) to trace error paths. **Snapshotting:** Period point-in-time state captures and error-stat logging. **Other Techniques:** Runtime query log parsing, automated anomaly detection/profiling, an time-travel debugging
Change management & impact Analysis	Prevent breaking changes across dependent systems.	**Proactive Schema Updates:** View downstream tables, reports, and APIs before altering schemas. **System Migration:** Safely migrate legacy data	**Visualization:** Downstream blast-radius graph views an column-level dependency graphs. **Snapshotting:** Pre-migration and schema version snapshots for regression comparisons. **Other Techniques:** Automated CI/CD schema validation, static code/AST

		warehouses or lakehouses.	parsing (SQL/dbt), an breaking change notifications/webhook
Privacy, security & compliance	Audit data flows and enforce data governance mandates. Be able to “prove” the effectiveness of a governance or a compliance plan.	**Sensitive Data / PII Tracking:** Monitor PII propagation across data stores. **Regulatory Auditing:** Satisfy GDPR, CCPA, BCBS 239, and HIPAA requirements.	**Visualization:** Flow maps showing data cross-boundary transfers and access points. **Snapshotting:** Immutable periodic compliance snapshots and historical state point-in-time audits. **Other Techniques:** Policy-tag propagation (inheritance), dynamic masking, data classification scanners and tombstone propagation for deletion audits (e.g., Right to be Forgotten)
AI & ML Model Governance	Ensure reproducibility, training provenance, and model safety.	**Training Data Provenance:** Trace model artifacts and weights back to raw data versions. **Model Drift & Debugging:** Track which	**Visualization:** End-to end ML pipeline graph (Dataset → Feature Store → Experiment → Model Registry). **Snapshotting:** Exact dataset version snapshots (e.g., DVC, LakeFS) and model weights/metadata snapshotting.

	models and serving endpoints depend on retraining jobs.	**Other Techniques:** Feature store lineage tagging, training pipeline orchestration telemetry, and data/concept drift monitoring.

Figure 2-x: use cases for lineage tracking

This also brings up the importance of the temporal dimension of lineage. The more sophisticated solutions track lineage across time—tracking not only what the current inputs to a dashboard are but also what those inputs were in the past, and how the landscape has evolved.

Until now we have described lineage as data-to-data: a dashboard traced back through transformations to its sources. AI adds two new kinds of edge to this graph. The first is *data-to-model*: which datasets, at which versions, were used to train or fine-tune a given model. The second is *data-to-prompt* (and *data-to-output*): which documents were retrieved into a model's context to produce a particular answer. Both edges now need to be tracked. Model-development and experiment-tracking tooling exists in part to record the data-to-model edge, and retrieval systems increasingly log which sources informed each response, so that an answer can be explained and audited the way the loan decision above could be.

The data-to-model edge also exposes a hard governance problem: machine *unlearning*. When sensitive or copyrighted content is accidentally pulled into a training set, it does not sit in a row you can delete—it is diffused into the model's weights, where the standard deletion tools described later in this chapter have no purchase. A model that has memorized a person's data cannot simply "forget" it on request, which puts model training on a collision course with the *right to be forgotten* enshrined in regulations like GDPR. Removing the influence of specific training data can demand costly retraining or specialized unlearning techniques whose guarantees are still

maturing. The practical conclusion is the one we reached for poisoning and bias: the only reliable control is prevention. Govern what enters the training pipeline in the first place—because once data has been learned, lineage can tell you it is in there but cannot easily get it out.

WHEN LINEAGE ESCAPES THE ORGANIZATION: THE CASE OF ELIAS THORNE

Lineage is hard enough inside one organization. Across the AI ecosystem it can become nearly impossible to trace—and a strange episode from 2026 shows why. Researchers and journalists noticed that when asked for a short story, a wide range of chatbots from different companies kept inventing the same character: Elias Thorne, usually a lighthouse keeper, usually with a backstory of some quiet coastal tragedy. A Cornell study sampled roughly 20,000 AI-generated stories and found that a mere eleven words—including “lighthouse,” “keeper,” and the names Elias, Mara, and Elara—appeared in about 88% of them, with “Elias the lighthouse keeper” turning up in roughly two-thirds. The researchers traced the pattern not to a flood of lighthouse novels in the pre-training data but to the alignment stage, where models are tuned toward “safe” content, and to a shared dataset in which only a tiny fraction of conversations carried the motif. The pattern then spread, because newer models are increasingly trained on text generated by older ones. For more, see "Elias in the Lighthouse, Again? Diagnosing Low Diversity in LLM Stories” by Sil Hamilton and David Mimno (Cornell University, Dept. of Information Science).

For data governance, Elias Thorne is a parable about lineage at ecosystem scale. A faint artifact entered the collective training data of the industry, was amplified by alignment, propagated from model to model as outputs became the next generation’s inputs, and finally “broke containment” into self-published books, music listings, and videos—at which point no one could say with confidence where it came from or how to remove it. This is data-to-model-to-data lineage with the labels rubbed off, and it previews how synthetic, AI-generated content can quietly contaminate the very corpora we train on (the phenomenon often called model collapse). It is one more argument for governing training inputs and tracking provenance aggressively: what you cannot trace, you cannot correct.

Key Management and Encryption

One consideration when storing data in any kind of system is whether to store it in a plain-text format or whether to encrypt it. Data encryption provides another layer of protection (beyond protecting all data traffic itself), as only the systems or users that have the keys can derive meaning from the data. There are several implementations of data encryptions:

Data encryption where the underlying storage can access the key. This allows the underlying storage system to effect efficient storage via data compression (encrypted data usually does not compress well). When the data is accessed outside the bounds of the storage system—for example, if a physical disk is taken out of a data center—the data should be unreadable and therefore secure.

Data encryption where the data is encrypted by a key inaccessible to the storage system, usually managed separately by the customer. This provides protection from a bad actor within the storage provider itself in some cases but results in inefficient storage and performance impact.

Just-in-time decryption, where, in some cases and for some users, it is useful to decrypt certain data as it is being accessed as a form of access control. In this case, encryption works to protect some data classes (e.g., "customer name") while still allowing insights such as "total aggregate revenues from all customers" or "top 10 customers by revenue," or even identifying subjects who meet some condition, with the option to ask for de-masking of these subjects later via a trouble ticket.

The major cloud platforms encrypt customer data by default, both in transit and at rest, so that customer data is protected even from low-level infrastructure access. Customers who need more control can manage or supply their own keys—for example, customer-managed encryption keys (CMEK) using a service such as Cloud KMS, or opting for customer-supplied encryption keys (CSEK) when they need more control over their data.

To provide the strongest protections, your encryption options should be native to the cloud platform or data warehouse you choose. The big cloud platforms all have a native key management that usually allows you to perform operations on keys without revealing the actual keys. In this case, there are actually two keys in play:

A data encryption key (DEK)

This key is used to directly encrypt the data by the storage system.

A key encryption key (KEK)

This key is used to protect the data encryption key and resides within a protected service, a key management service.

A Sample Key Management Scenario

In the scenario depicted in Figure 1-5, the table on the right is encrypted in chunks using the plain data encryption key. The data encryption key is not stored with the table but instead is stored in a protected form (wrapped) by a striped key encryption key. The key encryption key resides (only) in the key management service.

[Note: specific encryption topics and generally protection of data at rest is a broad topic; a good starter book would be Applied Cryptography by Bruce Schneier].

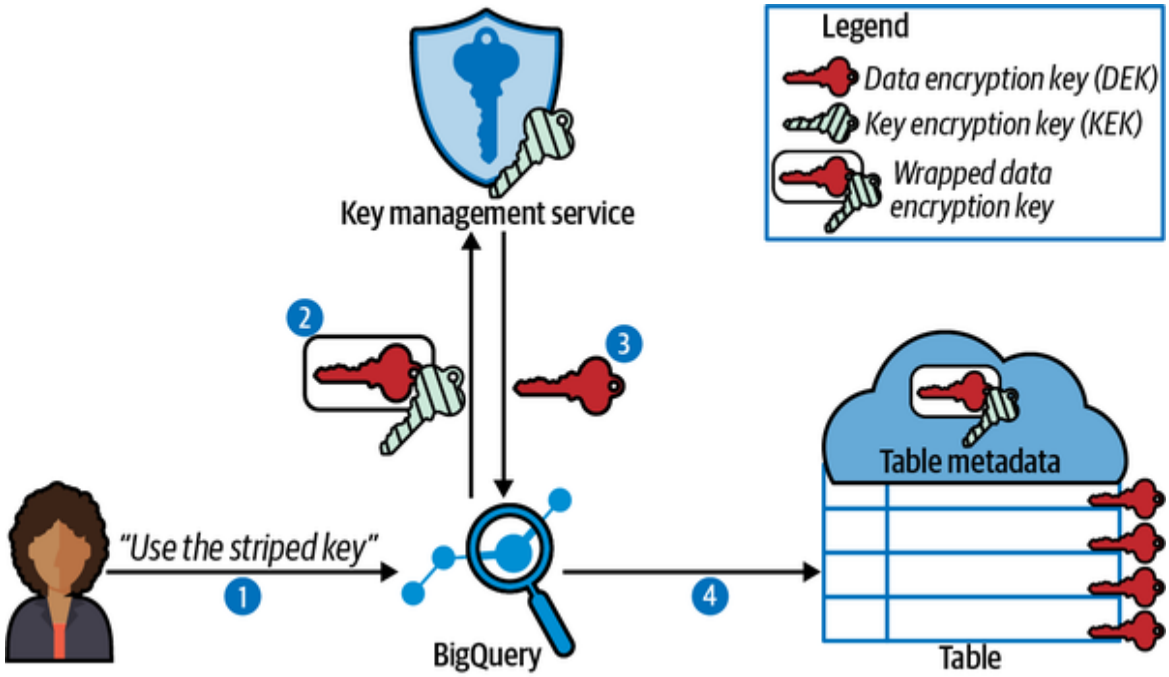


Figure 1-5. Key management scenario

To access the data, a user (or process) follows these steps:

The user/process requests the data, instructing the data warehouse (BigQuery) to use the striped key to unwrap the data encryption key, basically passing the key ID.

BigQuery retrieves the protected DEK from the table metadata and accesses the key management service, supplying the wrapped key.

The key management service unwraps the data protection key, while the KEK never leaves the vault of the key management service.

BigQuery uses the DEK to access the data and then discards it, never storing it in a persistent manner.

This scenario ensures that the key encryption key never leaves a secure separate store (the KMS) and that the data encryption key never resides on disk—only in memory, and only when needed.

Synthetic Data and Privacy-Enhancing Technologies

The tools in the previous section protect data by controlling who holds the keys. But a great deal of modern work—especially training and testing machine learning models—needs large volumes of realistic data, and for that use case encryption is awkward: you cannot train a model on ciphertext, and decrypting everything recreates the exposure you were trying to avoid. A family of approaches known as privacy-enhancing technologies (PETs) addresses this gap, letting organizations extract value from sensitive data while limiting exposure of the underlying records.

One of the most prominent of these is *synthetic data generation*. Instead of masking or encrypting real records, a generative model learns the statistical structure of a dataset and produces entirely new records that preserve its patterns and correlations but correspond to no real individual. Done well, a model trained on synthetic data performs nearly as well as one trained on the original, with far less privacy risk. A maturing market supports this, spanning commercial synthetic-data platforms (for example mostly.ai and tonic.ai)—including some aimed specifically at unstructured text—and open-source generation libraries such as the Synthetic Data Vault (SDV). There are even domain-specific platforms, such as Synthea, for generating realistic synthetic health records.

Synthetic data comes with an important caveat that belongs in any governance discussion: it is not automatically anonymous. A generator that overfits can memorize and reproduce real records, and membership-inference attacks can sometimes determine whether a specific person's data was in the training set. The formal protection that addresses this is *differential privacy*, a mathematical guarantee that the inclusion or exclusion of any single individual barely changes the output; the better synthetic-data tools can generate under a differential-privacy budget, trading a measurable amount of fidelity for a provable privacy bound. Without such a guarantee, *synthetic* should be treated as a risk-reduction measure, not a magic eraser, and the synthetic dataset's lineage back to its real source should be recorded like any other.

Two related PETs round out the toolkit. *Federated learning* trains a shared model across many devices or institutions without centralizing the raw data:

the data stays put (on your device), and only model updates move. And *confidential computing* uses hardware-based trusted execution environments (TEEs, or enclaves) to process data while it is decrypted in memory yet shielded even from the operator of the machine, extending protection to data in use, not just at rest and in transit. Chapter 7 returns to this family from the protection side, covering anonymization, differential privacy, and secure computation in depth.

Data Retention and Data Deletion

Another important item in the data governance tool chest is the ability to control how long data is kept—that is, setting maximal and minimal retention duration. There are clear advantages to identifying data that should survive occasional storage space optimization as being more valuable to retain, but setting a maximum amount of time on data retention for a less-valuable data class and automatically deleting it seems less obvious. Consider that retaining PII presents the challenges of proper disclosure, informed consent, and transparency. Getting rid of PII after a short duration (e.g., retaining location only while on the commute) simplifies the above.

One caveat now complicates the tidy picture of deleting data when its time is up. If a data class has been used to train a machine learning model, deleting the source records does not remove their influence from the model's weights—the deletion is incomplete in a way the "right to be forgotten" cares about (see the discussion of machine unlearning in the section "Lineage Tracking"). Two new artifact types also extend the reach of retention policy: the prompts users send to AI systems, which routinely contain personal data, and the outputs those systems return, which can inadvertently reproduce training data. Both should be brought under explicit retention and deletion rules rather than accumulating, unmanaged, in application logs.

When talking about data retention and data deletion, we're often thinking about them in the context of how to treat sensitive data—that is, whether to

retain it, encrypt it, delete it, or handle it some other way. But there are also scenarios around which your governance policies not only can save you from being out of compliance but also can protect you from lost work.

Although it's a bit old, an example that comes to mind around the subject of protection against data deletion is the accidental deletion of the movie *Toy Story 2* in 1998. During the film's production process, Oren Jacob, Pixar's CTO, and Technical Director Galyn Susman were looking at a directory that was holding the assets for several characters when they encountered an error: something along the lines of "directory no longer valid," meaning the location of those characters in the directory had been deleted. During their effort to walk back through the directory to find where the problem occurred, they witnessed, in real time, assets for several of the characters vanish before their eyes.

When all was said and done, an erroneous 10-character command had mistakenly deleted 90% of the movie. They were able to recover most of the movie, but unfortunately about a week's worth of work was lost forever. Find more about this incident in the excellent retelling: "How Pixar's Toy Story 2 was deleted twice, once by technology and again for its own good."

You can see from this example that even though sensitive data wasn't leaked or lost, a multitude of data was still accidentally deleted—some of which could never be recovered. We challenge you to consider in your governance program not only how you will deal with and treat sensitive data in terms of where or how long you retain it, and whether or not you delete it, but also how that same program can be implemented on other classes and categories of data that are important for you to back up. While a loss of data may not result in a breach in compliance, it could certainly result in other catastrophic business consequences that, if planned for, will hopefully not happen to you.

CASE STUDY: THE IMPORTANCE OF CLEAR DATA RETENTION RULES FOR DATA GOVERNANCE

A lesson about data retention was learned by a Danish taxi company that actually had a data governance policy in place.

To understand the story, we need a little bit of context about GDPR Article 5, the regulation covering treatment of European citizens' data by tech companies. GDPR details the standards that organizations have to follow when processing personal data. These standards state that data must be handled in a way that is transparent to its subject (the European citizen), collected for a specific purpose, and used only for that purpose. GDPR Article 5(1)(c) addresses data minimization by requiring that personal data be limited to what is necessary relative to the purpose for which it is processed. And Article 5(1)(e)—the one that's most relevant to the taxi company example—specifies that data cannot be stored any longer than is necessary for the purposes for which it was gathered.

The Danish taxi company in this example processed taxi ridership data for legitimate purposes, making sure there was a record of every ride, and of the associated fare, for example, for chargeback and accounting purposes.

The Danish taxi company, as mentioned, had a data retention policy in place: after two years, the data from a taxi ride was made anonymous by deleting the name of the passenger. However, that action did not make the data *completely* anonymous, as multiple additional details were (albeit transparently) collected. These included the geolocation of the taxi ride's start and end and the phone number of the passenger. With these details, even without the name of the passenger, it was easy to identify the passenger, and therefore the company was not in compliance with its own statement of anonymization; thus the data retention policy was actually not effective.

The lesson learned is that considering the business goal of a data retention policy, and whether or not that goal is actually achieved by the

policy set in place, is essential. In our case, the taxi company was fined by the European Union and was criticized for the fact that telephone numbers are actually used as unique identifiers within the taxi ride management system.

Workflow Management for Data Acquisition

One of the key workflows tying together all the tools mentioned so far is data acquisition. This workflow usually begins with an analyst seeking data to perform a task. The analyst, through the power of a well-implemented data governance plan, is able to access the data catalog for the organization and, through a multifaceted search query, is able to review relevant data sources. Data acquisition continues with identifying the relevant data source and seeking an access grant to it. The governance controls send the access request to the right authorizing personnel, and access is granted to the relevant data warehouse, enforced through the native controls of that warehouse. This workflow—identifying a task, shopping for relevant data, identifying relevant data, and acquiring access to it—constitutes a data access workflow that is safe. The level of data access requested—that is, access to metadata for search, access to the data itself, querying the data in aggregate—these are all data governance decisions.

Increasingly, the "analyst" at the start of this workflow is not a person at a keyboard but an automated pipeline or an AI agent acquiring data on a person's behalf. The workflow does not change shape, but it sharpens the attribution question we flagged earlier: the governance controls must record not just that access was granted, but which human principal the requesting service was acting for. We turn to that next.

IAM—Identity and Access Management

When talking about data acquisition, it's important to detail how access control works. The topic of access control relies on user authentication and,

per the user, the authorization of the user to access certain data and the conditions of access.

The objective of authenticating a user is to determine that you are who you say you are. Any user (and, for that matter, any service or application) operates under a set of permissions and roles tied to the identity of a service. The importance of securely authenticating a user is clear: if one user can impersonate a different user, there is a risk of assuming that user's roles and privileges and breaking data governance.

Authentication used to be traditionally accomplished by supplying a password tied to the user requesting access. This method has the obvious drawback that anyone who has somehow gained access to the password can gain access to everything that user has access to. Nowadays, proper authentication requires the following:

Something you know

> This will be your password or passphrase; it should be hard to guess and changed regularly.

Something you have

> This serves as a second factor of authentication. After providing the right passphrase, a user will be prompted to prove that they have a certain device (a cell phone able to accept single-use codes, a hardware token), adding another layer of security. The underlying assumption is that if you misplace that "object" you would report it promptly, ensuring the token cannot be used by others.

Something you are

> Sometimes, for another layer of security, the user will add biometric information to the authentication request: a fingerprint, a facial scan, or something similar.

Additional context

Another oft-used layer of security is ensuring that an authenticated user can access only certain information from within a specific sanctioned application or device, or other conditions. Such additional context often includes the following:

Being able to access corporate information only from corporate hardware (sanctioned and cleared by central IT). This, for example, eliminates the risk of "using the spouse's device to check for email" without enjoying the default corporate anti-malware software installed by default on corporate hardware.

Being able to access certain information only during working hours—thus eliminating the risk of personnel using their off-hours time to manipulate sensitive data, maybe when those employees are not in appropriate surroundings or are not alert to risk.

Having limited access to sensitive information when not logged in to the corporate network—when using an internet café, for example, and risking network eavesdropping.

The topic of authentication is the cornerstone of access control, and each organization will define its own balance between risk aversion and user-authentication friction. It is a known maxim that the more hoops employees have to jump through in order to access data, the more these employees will seek to avoid complexity, leading to shadow IT and information siloing—directions opposed to data governance (data governance seeks to promote data access to all, under proper restrictions). There are detailed volumes written on this topic, such as Identity and Access Management.

User Authorization and Access Management

Once the user is properly authenticated, access is determined by a process of checking whether the user is authorized to access or otherwise perform an operation on the data object in question, be it a table, a dataset, a pipeline, or streaming data.

Data is a rich medium, and sample access policies can have the following purposes:

- For reading the data directly (performing a `SELECT` SQL statement on a table, reading a file).
- For reading or editing the metadata associated with the data. For a table, this would be the schema (the names and types of columns, the table name). For a file, this would be the filename. In addition, *metadata* also refers to the creation date, update date, and "last read" dates.
- For updating the content, without adding new content.
- For copying the data or exporting it.
- There are also access controls associated with workflows, such as performing an extract-transform-load (ETL) operation to move and reshape the data (replacing rows/columns with others).

We have expanded here on the policies previously mentioned for data classes, which also detail partial-read access—which can be its own authorized function.

It's important to define identities, groups, and roles and assign access rights to establish a level of managed access.

Identity and access management (IAM) should provide role management for every user, with the capability to flexibly add custom roles that group together meaningful permissions relevant to your organization, ensuring that only authorized and authenticated individuals and systems are able to access data assets according to defined rules. Enterprise-scale IAM should also provide context (IP, device, the time the access request is being generated from, and, if possible, use case for access). As good governance results in context-specific role and permission determination before any data access, the IAM system should scale to millions of users, issuing multiple data access requests per second.

Non-Human Identity and On-Behalf-Of Access

Everything in our discussion so far assumes the identity being authenticated and authorized is a person. That assumption no longer holds. The fastest-growing population of identities in most organizations is non-human: service accounts, workloads, and now AI agents, each of which needs to be authenticated, scoped to least privilege, and given short-lived rather than standing credentials. Mechanisms such as *workload identity federation* let a workload in one cloud assume a tightly scoped identity in another without long-lived secrets—which matters, because these non-human identities have become a common path for breaches.

The subtler problem is the one we have flagged twice already: on-behalf-of (OBO) access. When an AI assistant retrieves data for a user—the core move in RAG—whose permissions apply? If the system enforces the permissions of the service account that runs the assistant, it becomes a confused deputy: a powerful, broadly privileged identity that will happily fetch and surface documents the actual user was never allowed to see. The governance requirement is that retrieval enforces the end user's permissions, evaluated at query time, not the service's. In practice this means the access controls have to reach into the vector store and document index themselves — filtering candidate results by the user's entitlements before they are assembled into the model's context — and it means the system must reliably attribute each action back to the human principal who triggered it, preserving that link through every hop the agent makes.

This attribution-to-a-human is the bridge from classical IAM to AI agent governance. An agent that can chain actions across systems — reading here, writing there, calling a third tool — multiplies both its usefulness and its blast radius. Governing it well takes the same primitives we have described, applied to non-human actors: a real identity, scoped and time-bound authorization, a full audit of what it did and on whose behalf, and the ability to revoke it instantly. Because agents and pipelines routinely span clouds, these controls are best expressed in cloud-agnostic terms—defined against identities and data assets rather than re-implemented inside each provider's

native IAM—so that a single policy follows the principal wherever the work runs.

Summary

In this chapter, we have gone through the basic ingredients of data governance: the importance of having a policy book containing the data classes managed, and how to clean up the data, secure it, and control access. To those classic ingredients the AI era has added several more: governing unstructured data and the vector embeddings derived from it, defending data quality against deliberate poisoning, extending lineage from data-to-data into data-to-model and data-to-prompt, generating synthetic data under privacy guarantees, and granting identity and scoped authorization to the non-human agents that increasingly act on our behalf. Now it is time to go beyond the tooling and discuss the importance of the additional ingredients of people and processes to a successful data governance program.

Tools and tooling alone will not solve data governance. In the next chapter, we turn to the people and processes that are essential parts of a well-rounded data governance strategy.

Chapter 2. Ingredients of Data Governance: People and Processes

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. Please note that the GitHub repo will be made active later on.

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

As mentioned in the preceding chapters, companies want to be able to derive more insights from their data. They want to make "data-driven decisions." Gone are the days of business decisions being based exclusively on intuition or observation. Big data and big data analytics now allow for decisions to be made based on collecting data and extracting patterns and facts from that data.

We have spent much time explaining how this movement into using big data has brought with it a host of considerations around the governance of that data, and we have outlined tools that aid in this process. Tools, however, are not the only factor to evaluate when designing a data governance strategy—the *people* involved and the *process* by which data governance is implemented are key to a *successful* implementation of a data governance strategy.

The people and process are aspects of a data governance strategy that often get overlooked or oversimplified. There is an exceedingly heavier reliance on governance tools, and though tools are getting better and more robust, they are not enough by themselves; *how* the tools are implemented, an understanding of the people using them, and the process set up for their proper use are all critical to governance success as well.

AI has become part of the governance picture because of the emergence of human annotators and AI agents whose actions must be governed. Agentic AI can take on governance roles as well, taking a first pass at classification, answering business users' questions directly, and enforcing policy across clouds.

The People: Roles, Responsibilities, and Hats

Many data governance frameworks revolve around a complex interplay of many roles and responsibilities. These frameworks rely heavily on each role playing its part in keeping the well-oiled data governance machine running smoothly.

The problem with this is that most companies are rarely able to exactly or even semi-fully match these frameworks, due to lack of employee skill set or, more commonly, a simple lack of headcount. For this reason, employees working in the information and data space of their company often wear different user *hats*. We use the term *hat* to delineate the difference between an actual *role or job title* and *tasks* that are done. The same person can perform tasks that align with many different roles or wear many different hats as part of their day-to-day job.

User Hats Defined

In Chapter 1, we outlined three broad categories of governors, approvers, and users. Here we will look in depth at the different hats (versus roles) within each category (and expand on an additional category of ancillary

hats), the tasks associated with each hat, and finally, the implications and considerations when looking at these hats from a task-oriented perspective versus a role-based approach. In Table 3-1 we've listed out each hat with its respective category and key tasks for quick reference, with more detailed descriptions following.

Table 2-1. Table 3-1. Different hats with their respective categories and the tasks associated with them

Hat	Category	Key tasks
Legal	Ancillary	Knows of and communicates legal requirements for compliance
Privacy tsar	Governor	Ensures compliance and oversees company's governance strategy/process
Data owner	Approver (can also be governor)	Physically implements company's governance strategy (e.g., data architecture, tooling, data pipelining, etc.)
Data steward	Governor	Performs categorization and classification of data
Data analyst/data scientist	User	Runs complex data analytics/queries
Business analyst	User	Runs simple data analyses
Customer support specialist	Ancillary (can also be a user or an AI agent)	Views customer data (but does not use this data for any kind of analytical purpose)

Hat	Category	Key tasks
AI annotator	AI agent or User	Labels training data, evaluates model outputs, and supplies reinforcement-learning feedback
AI agent (on-behalf-of)	AI agent as User	Executes tasks under a specific human's delegated identity and scope
Autonomous AI agent	AI agent as Approver	Acts on its own system-level credentials, with no human principal
C-suite	Ancillary	Funds company's governance strategy
External auditor	Ancillary	Audits a company's compliance with legal regulations

Legal (ancillary)

Notwithstanding the title of *legal*, this hat may or may not be an actual attorney. This hat includes the tasks of ensuring the company is up to date in terms of compliance with the legal requirements for its data handling and communicating this information internally. Depending on the company, the person with this hat may actually be an attorney (this is especially true for highly regulated companies, which will be covered in depth later) who must have a deep knowledge of the type of data collected and how it's treated to ensure that the company is in compliance in the event of an external audit. Other companies, especially those dealing with sensitive but not highly regulated data, are more likely to simply have someone whose task is to be up to date on regulations and have a deep understanding of which ones apply to the data the company collects.

Privacy tsar (governor)

We chose to title this hat *privacy tsar* because this is a term we use internally at Google, but this hat has also been called *governance manager*, *director of privacy*, and *director of data governance* in other literature. The key tasks of this hat are those which ensure that the regulations the legal department has deemed appropriate are followed. Additionally, the privacy tsar also generally oversees the entire governance process at the company, which includes defining which governance processes should be followed and how. We will discuss other process approaches later in this chapter. It's important to note that the privacy tsar may or may not have an extremely technical background. On the surface it might seem that this hat would come from a technical background, but depending on the company and the resources it has dedicated to its data governance efforts, these tasks are often performed by people who sit more on the business side of the company rather than on the technical side.

Understanding the movement of people is of utmost importance when it comes to battling epidemics and pandemics like COVID-19. Google, a company that processes significant amounts of highly personal data that includes location information, was torn between helping healthcare providers and authorities to battle the deadly pandemic more effectively and preserving the trust of the billions of people worldwide who use Google's services.

Privacy tsar, work example 1: community mobility reports

The challenge of preserving privacy while at the same time providing useful, *actionable* data to health authorities required the full attention of Google's privacy tsars, the people entrusted with creating the internal regulations that make sure that technology does not intrude into users' personal data and privacy.

The solution they found was to provide information in an aggregated form, based on anonymized sets of data from only those users who have turned on "location history" in Google's services. This setting is off by default, and users need to "opt in" to enable it. Location information history can always

be deleted by the user at any time. In addition, differential privacy (a technique covered in Chapter 7) was further used to identify small groups of users with outlier results and eliminate those groups completely from the provided solution. Another differential privacy technique was employed to add statistical noise to the results; the noise, statistically irrelevant for aggregates, helps ensure that no individual can be tracked back through the data.

The result was a useful set of reports tracking communities and changes in behavior over time. Health officials can then assess whether "stay at home" orders are being complied with, and trace back sources of infection due to people congregating.

In Figure 2-1, you see the results of that work. A sample from the report for San Francisco County from 2022 showed increased people presence in residential areas (bottom right) but a reduced presence across the board in retail sites, grocery stores, parks, transit stations, and workplaces. Note that no individual location is named, yet the data was useful for health officials in estimating where people congregate. For example, an order about the reopening of retail stores could be considered.

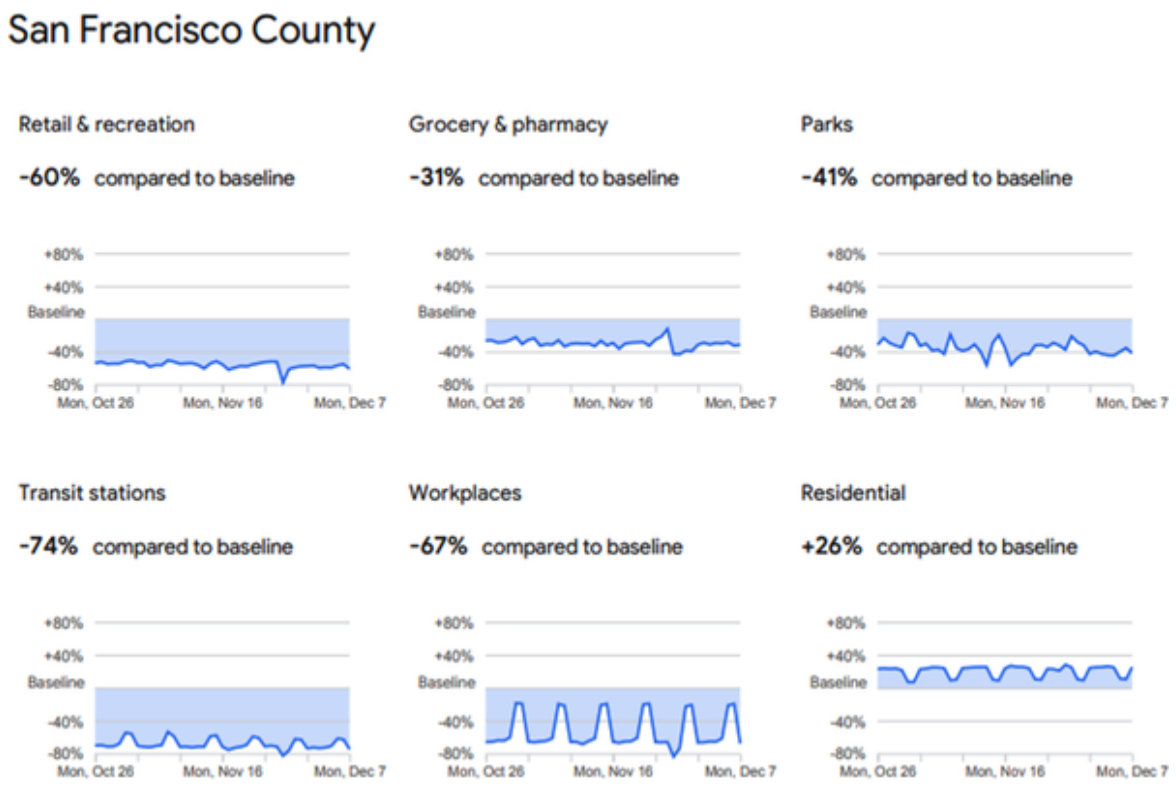


Figure 2-1. Example from the Google mobility reports

Privacy tsar, work example 2: Exposure notifications

An even more daunting task related to COVID-19 was how to safely (from a privacy perspective) inform people about prolonged exposure to a person diagnosed with COVID-19. Because such a virus is highly contagious and can be transmitted through the air, identifying exposures and making sure people who were inadvertently exposed to a positively diagnosed person get tested (and isolate themselves if testing positive), is crucial to breaking infection chains and limiting outbreaks. This process is a recognized technique in battling infections and is otherwise known as *contact tracing*. Technology can augment this technique by immediately alerting the individual as an alternative to a prolonged phone investigation in which public health authorities question a positively diagnosed individual as to their whereabouts over the incubation period. (Many people cannot accurately identify where they have been over the course of the past few days, nor can everyone easily produce a list of all the people they have interacted with over the course of the past week.)

However, a positive COVID-19 diagnosis is highly personal, and having this information delivered to everyone that a diagnosed person came in contact with is an emotionally loaded process. Furthermore, having your technology do that for you is highly intrusive and will cause resistance to the point of not enabling this technology. So how does a privacy tsar thread the needle between preserving personal information and privacy and combating a deadly disease? The solution that was found maintains the principles required to ensure privacy:

- It must be an *opt-in* solution—the people must enable it, and the product provides information to ensure consent is acquired after being informed.
- Since the topic is whether or not the subject was next to a diagnosed person, location information, though it might be useful to health authorities to understand where the incident occurred, is not collected. This is a decision made by the privacy tsar in favor of preserving privacy.
- The information is shared only with public health authorities and not with Google or Apple.

So how does the solution work? Every phone beams a unique yet random and frequently changing identifier to all nearby phones; the phones collect the list of beacons, and this list is matched with anyone who has reported their diagnosis. If your phone was in proximity to the phone of someone who has uploaded a positive diagnosis, the match will be reported to you (see Figure 2-2). Note that the identity of the infected individual is not reported, nor is the specific time and place. Thus the crucial information (you have been near a positively diagnosed individual so get tested!) is shared without sacrificing the privacy of the infected individual.

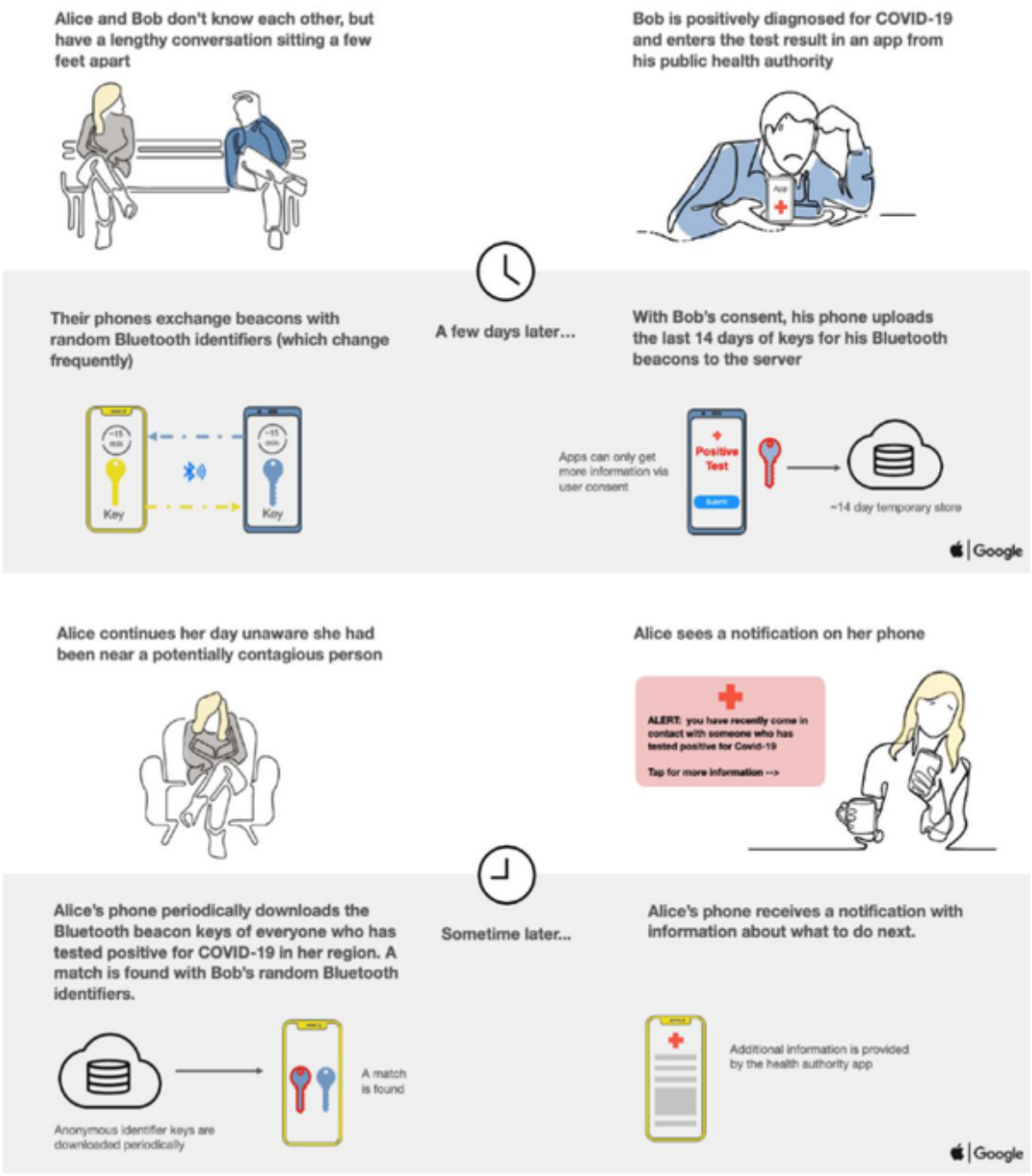


Figure 2-2. Excerpt from the Google/Apple guide to the exposure notification technology

These privacy principles must also account for the potential use of this data to train models. A privacy tsar must govern how models ingest, retain, and process personal location data and contact-tracing identifiers—ensuring that opt-in location history is not silently absorbed into a training corpus, that ephemeral contact-tracing beacons are not retained past the purpose that justified them, and that the differential-privacy guarantees described above

are preserved end to end rather than stripped away the moment the data crosses into a training pipeline. Nor need the differential-privacy work described above rest entirely on human governors. Automated machine-learning tooling calibrates and injects the statistical noise dynamically, enforces a privacy budget across queries, and holds the guarantee constant as the underlying data shifts—taking the operational burden off the privacy tsar while leaving the policy decision (how much privacy, at what cost to utility) where it belongs, with a human.

The AI mandate reaches past the training pipeline to the agents that act once a model is deployed. An agentic pipeline reads, joins, and acts on data with no human in the loop. That is an unforgiving surface. A single over-scoped credential or an unlogged action can cause a catastrophic exposure, and it tends to surface only after the fact. Auditing these pipelines thus falls to the privacy tsar—confirming that each agent runs within a least-privilege scope, that its actions trace to a named owner, and that its access to sensitive data is logged and reviewed as rigorously as any human's.

Data owner (approver/governor)

In order for the privacy tsar's governance strategy/process to be realized, the data owner is needed.

> While "classical literature" on data governance often separates data owners and data custodians (the former residing more on the business side of things, and the latter more on the technical side), during the course of our research and interviews with many companies we found that, in practice, these two "sides of the coin" are often conflated, and the actual tasks of data ownership tend to fall on those with technical expertise.

The tasks of the data owner include physically implementing the processes and/or strategies laid out by the privacy tsar. This most often includes the ideation and creation of the data architecture of the company, along with choosing and implementing tooling and data pipeline and storage creation,

monitoring, and maintenance—in other words, "owning the data." It's clear from the task descriptions that these must be performed by someone with quite a bit of technical background and expertise; hence people who wear the data owner hat largely are engineers or folks with an engineering background.

DAY IN THE LIFE

A data owner spends the morning approving a request to add a new sensitive column to the customer table, ruling that it must inherit the table's existing access policy rather than a more permissive one. They neither grant the access themselves nor run the backup; they decide what the policy is and remain accountable for the data. This is the boundary most often blurred in practice: the owner is accountable *for* the data, not necessarily the person who *touches* it.

Data steward (governor)

When researching data governance, you will find that there is likely to be much weight given to the role of *data steward*, and that's with good reason—the tasks of a data steward include categorization and classification of data. For any sort of governance to be implemented, data must be defined and labeled to clearly identify what the data is: sensitive, restricted, health related, and so on. A large part of the reason we advocate the usage of the term *hats* versus *roles* is exemplified by the fact that it's very rare to find a singular person doing the "role" of data steward in the wild. The act of data "stewardship" is highly, highly manual and extremely time consuming. In fact, one company we spoke to said that for a short while it actually had a "full time" data steward who quit after several months, citing the job as being "completely soul sucking." Because of the manual, time-consuming nature of the role—coupled with the fact that in most cases there is no *dedicated person* to perform stewardship duties—these duties often fall to many different people across the company or to someone who has another role/other duties they perform as well. As such, full data categorization/classification is often not done well, not done fully, or, in the worst cases, just not done at all.

This is an important item to note, because without stewardship, governance is incomplete at best. We will cover this in more depth later, but here is where we begin to see a recurring theme when looking at the people and the process. Many of the governance processes that most companies employ right now are undertaken to *get around* the fact that stewardship falls short. As we outlined in Chapter 1, the quick growth and expansion of data collected by companies has resulted in an overwhelm of data, and companies simply don't have the time or headcount to dedicate to being able to categorize, classify, and label ALL of their data, so they create and utilize other methods and strategies to "do their best given the limitations."

> **DAY IN THE LIFE**
>
> A steward reviews a newly ingested dataset, tags three columns as health-related and one as restricted, and corrects a misclassification that would otherwise have exposed location data to the general analytics tier—this is a few minutes of work whose absence would quietly defeat every control the owner and custodian put in place.

Generative AI copilots can take a first pass at this soul-sucking work. A copilot scans a newly landed table, proposes classifications (sensitive, restricted, health-related), drafts column descriptions, and flags likely personally identifiable information (PII) for a human to confirm—turning enrichment from an all-or-nothing manual slog into a review-and-approve task. The steward's judgment still governs; only the marginal cost of cataloging the next thousand columns falls toward zero.

Data analyst/data scientist (user)

Data analysts and data scientists are, in general, some of the primary or key users of data within a company and are largely who data governance efforts are for. Companies struggle with governance and security of their data versus the democratization of their data. In order to be data driven, companies must collect and analyze large amounts of data. The more data that can be made available to analysts and scientists, the better—unless of course that data is sensitive and should have limited access. Therefore, the

better the governance execution, the better (and the more safely) analysts or scientists are able to do their job and provide valuable business insights.

Business analyst (user)

While data analysts and data scientists are the main users or consumers of data, there are a few people in the periphery of a company who also use and view data. In moving toward being more data driven, companies have folks who sit on the business side of things who are very interested in the data analyses produced by analysts and scientists. In some companies, data engineers aid in the creation and maintenance of much simpler analytics platforms to aid in "self-service" for business users. More and more business people in companies have questions that they hope crunching some data will help them to answer. Analysts/scientists thus end up fielding many, many inquiries, some of which they simply don't have time to answer. By enabling business users to directly answer some of their own questions, analysts/scientists are able to free up their time to answer more complex analysis questions.

A natural-language self-service layer eases this bottleneck further. A business user can ask "what was net revenue by region last quarter?" and have the interface generate, run, and explain the query, subject to the same access policy a human would face. The scientists are freed for the genuinely hard questions, and the governance controls travel with the query rather than being bypassed by it.

Customer support specialists (user/ancillary)

While a customer support specialist is technically only a *viewer* of data, it's worth noting that there *are* folks with this role who will need access to some sensitive data, even though they don't have any tasks around manipulating that data. In terms of hats, customer support specialists *do* tend to have this as their sole role and are not also doing other things; however, they are consumers of data, and their needs, and how to grant them appropriate access, must be considered and managed by the other hats

executing a company's governance strategy. Customer support is often triaged or carried out by AI agents.

AI annotator (AI agent or user)

Modern pipelines increasingly depend on people who do not analyze data so much as teach models from it. The AI annotator labels raw training data, rates and rewrites model outputs for reinforcement learning from human feedback (RLHF), and produces the high-quality metadata that supervised and preference-based training require. They are users, but unusual ones: their work demands broad exposure to raw, frequently sensitive data, which makes their access scope a first-order governance concern rather than an afterthought.

Agentic AI identities (AI agent or user)

An AI agent is a non-human actor, and it forces a question that never arose for a human analyst: *whose authority is the agent exercising?* Two cases must be governed differently. An *on-behalf-of agent* acts under a delegated human identity and must inherit that human's scope exactly—an agent acting for user A should be able to see only the orders belonging to user A, never user B's, even though the underlying model is technically capable of reading both. An *autonomous agent* holds its own system-level credentials and acts with no human principal; it requires its own least-privilege scope, its own audit trail, and a named human owner who remains accountable for what it does. Conflating the two—letting an on-behalf-of agent quietly fall back on system credentials, or running an autonomous agent under a borrowed human login—is how scope violations and unattributable actions creep in.

C-suite (ancillary)

In many companies, members of the C-suite have limited tasks in relation to the actual execution of a data governance strategy. They are nonetheless a critical hat in the grand scheme of governance because they hold the purse strings. As we mentioned earlier, tools and headcount are critical factors when considering a successful data governance strategy. It thus makes sense

that the person who actually *funds* that strategy must understand it and be on board with the funding that makes it a reality.

External auditor (ancillary)

We have included the hat of external auditor in this section despite the fact that external auditors are not within a particular company. Many of the companies we've spoken to have mentioned the importance of the external auditor in their governance strategy. No longer is it good enough to simply be "compliant" with regulations—companies now often need to *prove* their compliance, which has direct implications for the way a governance strategy and process is employed. Oftentimes, companies need to prove who has access to what data as well as all the different locations and permutations of that data (its lineage). While internal tool reporting can generally help with providing proof of compliance, the way that a governance strategy is set up and tended to can help, or hinder, the production of this documentation.

Data Enrichment and Its Importance

As we mentioned at the beginning of this section, one person may wear many hats—that is, perform many of the tasks involved in carrying out a data governance strategy at a company. As can be seen in the list of hats and the (fairly long) list of tasks *within* each hat shown in Table 2-1, it's easy to see how many of these tasks may not be completed well, if at all.

While there are many tasks that are important to successful implementation of a data governance strategy, it could be argued that the most *critical* are data categorization, classification, and labeling. As we pointed out, these tasks are manual and highly time consuming, which means that they rarely get executed fully. There is a saying: "In order to govern data, you must first know what it is." Without proper data *enrichment* (the process of attaching metadata to data), the central task of the data steward hat, proper data governance falls short. This central task of the data steward is so key, however, that what we commonly see is that the person who wears the data steward hat also often wears the privacy tsar hat as well as the data owner

hat, and/or they may even wear a hat in a completely different part of the company (a common one we see is business analyst). Wearing many hats results in a limited amount of tasks that can be done (one person can only do so much!), and most often the majority of the time-consuming task of data enrichment falls off the list.

In the next section we will cover some common governance processes we've observed over the years. One thing to keep in mind while reading through the processes is how the hats play a role in the execution of these processes. We will discuss their interplay later in this chapter.

The Process: Diverse Companies, Diverse Needs and Approaches to Data Governance

It's important to note that in the discussion of the people and processes around data governance, there is no "one size fits all" approach. As mentioned, in this section we will begin to examine some broad company categories, outline their specific concerns, needs, and considerations, and explore how those interplay with a company's approach to data governance.

> **NOTE**
>
> We'd like to reemphasize the point that there is no "one size fits all" approach to governance; and you need to fit your program to your needs—whether those be mitigated by headcount, the kind of data you collect, your industry, etc. We have seen many governance approaches and frameworks that are somewhat inflexible and thus might be difficult to implement if you don't fit their particular parameters. We hope that through exploration of elements to consider you're able to create a framework for yourself that not only matches your governance needs and goals but also matches where your company is right now, and where you'd like to be in the future.

Legacy

Legacy companies are defined as companies that have been around for quite some time and most certainly have, or have had, legacy on-premises (on-prem) systems—most often many different systems, which bring with them

a host of issues. The time and level of effort this work requires often leads to it not being done fully, or simply not being done at all. Further, for consistency (and arguably for the most effective use of data and data analytics), every company should have a *central data dictionary*, defining all data names, classes, and categories, that is standardized and used throughout the company. Many legacy companies lack this central data dictionary because their data is spread out through these various on-prem systems. More often than not, these on-prem systems and the data within them are associated with a particular branch or line of business. And that branch, in and of itself, is agnostic of the data that resides in the other lines of the business's systems. As such, there ends up being a dictionary for each system and line of business that may not align to any other line of business or system, which makes cross-system governance and analytics nearly impossible.

A prime example of this would be a large retailer that has a system that houses its online sales and another system that handles its brick-and-mortar sales. In one system the income the company receives from a sale is called *revenue*, while in the other system it's simply called *sales*. Between the two systems, the same enterprise dictionary is not being used—one can see where this becomes problematic if executives are trying to figure out what the total income for the company is. Analytics are nearly impossible to run when the same data does not have the same metadata vernacular attached. This becomes an even larger issue when considering sensitive data and its treatment. If there is no agreed-upon, company-wide terminology (as outlined in the enterprise dictionary), and if that terminology is not implemented for *all sources* of data, governance is rendered incomplete and ineffective.

The power of the cloud, big data analytics, and frontier AI models drives many companies to want to migrate their data, or a portion of their data, to the cloud—but the past pain of inconsistent enterprise dictionaries and haphazard governance gives them pause. They don't want to "repeat their past mistakes"; they don't want to simply replicate all of their current issues. While tools can help to right past wrongs, they simply aren't

enough. Companies need (and desire) a framework for how to move their data and have it properly governed from the beginning, with the right people working the right process.

Cloud Native/Digital Only

Cloud-native companies (sometimes referred to as *digital only*) are defined as companies who have, and have always had, all of their data stored in the cloud. Not always, but in general, these tend to be much younger companies who have never had on-premises systems and thus have never had to "migrate" their data to a cloud environment. Based on that alone, it's easy to see why cloud-native companies don't face the same issues that legacy companies do.

Despite the fact that cloud-native companies do not have to deal with on-prem systems and the "siloing" of data that often comes along with that, they still may deal with different clouds as well as different storage solutions within and between those clouds that have their own flavor of siloing. For one, a centralized data dictionary, as we've explored, is already a challenge, and having one that spans multiple clouds is even more daunting. Even if a centralized data dictionary is established, the process and tools (because some clouds require that only certain tools be used) by which data is enriched and governed in each cloud will probably be slightly different. And that is something that hinders consistency (in both process and personnel) and thus hinders effectiveness. Further, even within a cloud, there can be different storage solutions that also may carry with them different structures of data (e.g., files versus tables versus jpegs). These structures can be difficult to attach metadata to, which makes governance additionally difficult.

In terms of cloud—and data governance within clouds specifically—cloud-native companies tend to have better governance and data management processes set up from the beginning. Because of their relatively young age in dealing with data, there is often less data overall, and they have more familiarity with some common data-handling best practices. Additionally, cloud-native companies by definition have *all* of their data in the cloud,

including their most sensitive data—which means that they've most likely been dealing with the need for governance since the beginning, and they most likely have some processes in place for governing, if not *all* of their data, at least their most sensitive data.

Retail

Retail companies are an interesting category, as not only do they often ingest quite a bit of data from their own stores (or online stores), but they also tend to ingest and utilize quite a bit of third-party data. This presents yet another instance in which data governance is only as good as the process set up for it and the people who are there to execute it. The oft-mentioned importance of creating and implementing a data dictionary applies, as well as having a process around how this third-party data is ingested, where it lands, and how governance can be applied.

One twist that we have not discussed but that very much applies in the case of retail is that of governance beyond simple classification of data. So far we have discussed the importance of classifying data (especially sensitive data) so that it can be known and thus governed appropriately. That governance most often relates to the treatment of and/or access to said data. But data access and data treatment are not the only aspects to consider in some instances; the use case for that data is also important. In the case of retail, there may be a certain class of data—email for example—that was collected for the purpose of sending a customer their receipt from an in-store purchase. In the context (use case) of accessing this data for the purposes of sending a customer a receipt, this is completely acceptable.

If, however, one wanted to access this *same* data for the purpose of sending a customer some marketing material around the item they just purchased, this use case would *not* be acceptable unless the customer has given explicit consent for their email to be used for marketing. Now, depending on the employee structure at a company, this problem may not be solvable with simple role-based access controls (fulfillment gets access, marketing does not) if the same employee may cover many different roles. This warrants

the need for a more complex process that includes establishing use cases for data classes.

COMBO OF LEGACY AND RETAIL

A particularly interesting use case we've come across in our customer interviews is that of a very large legacy retail company. In business for over 75 years, this company is looking to leverage powerful data analytics tools to help it move toward being a more data-driven company.

The struggle, interestingly, is not only in its old legacy on-prem data-storage systems but also in its internal processes around data and data management.

It currently has its data separated into several data marts, a configuration aimed at distributing responsibility for segments of data: a mart for marketing for in-store sales, for third-party sales, and so on. This, however, has become problematic for the company, as not only is there no "central source of truth" in terms of an enterprise dictionary, but it also cannot run any kind of analytics across its data marts, resulting in duplication of data from one mart to another since analytics can only be run within a mart.

Historically, the company was very focused on keeping all of its data on-prem; however, this view has changed with the increased security that is now available in the cloud, and the company is now looking to migrate all of its data off-premises. Through this migration effort, not only is the company looking to change the basic infrastructure of its data story from a decentralized one (multiple marts) to a centralized one (centralized data warehouse), but it is also using this as an opportunity to restructure its internal processes around data management and governance (see Figure 2-3).

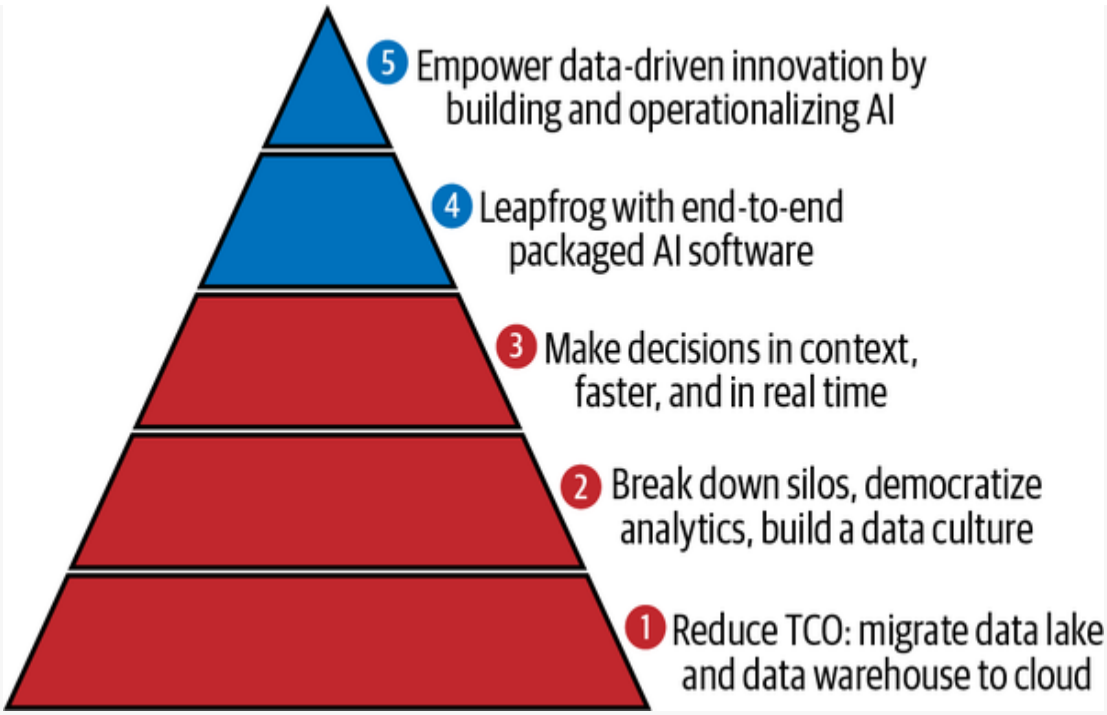


Figure 2-3. Breaking down data silos by restructuring internal processes is often a key stage after enterprises consolidate their on-premises data into the cloud

Data centralization enables the company to have one enterprise dictionary that allows for all new data—namely sensitive data—to be quickly marked and treated accordingly. Quick and easy handling of sensitive data also enables quicker and easier access controls (as they need to be applied only once, as opposed to each mart). This not only saves overhead effort but also allows for the implementation of more easy-to-use self-service analytics tools for employees who may not be analysts by trade but, with the right tools and access to data, are able to run simple analytics. This new process, however, is really only good for new incoming data.

This company, like many legacy companies with a lot of historical data, is struggling with how to handle the data it already has stored. It currently has 15 years of data (~25 terabytes) stored on-premises, and while it knows there is a wealth of information there, the time and effort required to migrate, enrich, and curate all this data seems daunting at best, especially when the payoff is not known.

Highly Regulated

Highly regulated companies represent the sector of companies that deal with extremely sensitive data—data that often carries additional compliance requirements beyond the usual handling of sensitive information. Some examples of highly regulated companies would be those dealing with financial, pharmaceutical, or healthcare services.

Highly regulated companies deal with multiple kinds of sensitive data. They have to juggle not only basic data governance best practices but also the additional regulations related to the data they collect and deal in, and they face regular audits to make sure that they are aboveboard and compliant. As a result of this, many highly regulated companies are more sophisticated in their data-governance process. As they've had to deal with compliance related to their data from the get-go, they often have better systems in place to identify and classify their sensitive data and treat it appropriately.

Also, for many of these kinds of companies, their business is based solely around sensitive data, so not only do they tend to have better processes in place from the beginning, but also those processes most often include a more well-funded and sophisticated organization of the people who handle that sensitive data. These companies tend to actually have people dedicated to each of the hats discussed earlier, and that additional headcount, as we pointed out, can be the deciding factor in whether a data-governance strategy is successful or unsuccessful.

One final note about highly regulated companies is that, as a result of the legal requirements they face on the handling of certain data, they can function similarly to legacy companies in that they have difficulty moving off-prem and/or trying out new tools. Any tool that will touch sensitive data under a regulation must meet the standards (fully) of that regulation. As an example, a tool or product in beta may be used by a healthcare company only if it is HIPAA compliant. Many product betas, because they're used for early access and as a way to work out bugs, are not designed to meet the most stringent compliance standards. While the final product may be compliant, the beta generally is not, meaning that highly regulated

companies often don’t get to experiment with these new tools/products and thus have trouble migrating to new, potentially better tools than the ones they’re currently using.

UNIQUE HIGHLY REGULATED ORGANIZATION: HOSPITAL/UNIVERSITY

When discussing highly regulated industries, finance and healthcare are the ones most commonly referenced. In our discussions with customers we came across another highly regulated industry that had some unique challenges: hospital/universities. These “companies” are unique in that they collect a lot of clinical data from their hospitals, but they also collect (and produce) a lot of research data through university-sponsored research studies.

Each of these types of data comes with its own specific regulations and standards for research—e.g., HIPAA covers clinical data, and the Institutional Review Board (IRB) protects the rights and welfare of human research subjects recruited to participate in research activities.

One particular hospital/university we spoke to was looking at the use case of being able to run secondary analytics across its clinical and research data. Currently it has two main pipeline solutions for its data: one for clinical and one for research.

For its clinical pipelines, data is stored on-prem in a Clarity data warehouse for each hospital, and only analysts with access to that database are able to run analytics.

For its research pipelines, each “lab” within the university has its own on-prem storage, and again, only analysts with access to that server can run analytics. It can be seen that not only does this structure not allow for secondary analyses to be run across labs, but they also can’t be run across clinical and research.

Knowing that there is a wealth of value in these secondary analytics, this hospital/university decided to migrate a large portion of its clinical and research data to the cloud, to get it all in one central location. In order to do this, however, much needs to be done to make the data comply with healthcare and research regulations. As such, the hospital/university created a specialized team dedicated to this

migration effort; that team's role included tasks such as: creating an enterprise dictionary, enriching data, reviewing the presence of sensitive data, reviewing policies attached to data, applying new policies to data, and applying a standardized file structure.

Fortunately for this organization, it was able to secure funding for such an elaborate undertaking, and yet it is still looking at ways to automate its process. Migration and the setup of data in the cloud is but one hurdle—maintaining and managing a data store going forward will require effort as well, and as such, tools that enable automation are top of mind for this organization.

Small Companies

For our purposes, we define a *small* company as one that has fewer than one thousand employees. One of the benefits of small companies is their smaller employee footprint.

Smaller companies often have small data-analytics teams, which means that there are fewer people who actually need to touch data. This means that there is less risk overall. One of the primary reasons to govern data is to make sure that sensitive data does not fall into the wrong hands. Fewer employees also makes for a much less arduous and less complicated process of setting up and maintaining access controls. As we discussed earlier, access controls and policy management can get increasingly complicated, especially when factors such as use case for data come into play.

Another benefit of a small amount of people touching data is that there is often less proliferation of datasets. Data analysts and scientists, as part of their jobs, create many different views of datasets and joins (combining tables from different dataset sources). This proliferation of data makes it hard to track where the data came from and where it's going to (not to mention who had and now has access). Fewer analysts/scientists mean fewer datasets/tables/joins, resulting in data that's much easier to track and thus easier to govern.

Large Companies

While there are many more company types, we will end our exploration of broad categories with *large* companies, defined as companies with more than one thousand employees.

If small companies have the governance benefit of dealing with less data, large companies deal with the reverse, in that they often deal with *a lot* of data. Large companies not only generate a great deal of data themselves but also often deal with a lot of third-party data. This results in immense difficulty in wrapping their arms around it all; they are overwhelmed by the data and often struggle to govern even a portion of it. As such, only some data gets enriched and curated, which means that only some of their data is able to be used to drive insights.

Large companies often put processes in place to limit this overwhelm by choosing only select data to enrich, curate, and thus govern. A common strategy for limiting the data that a data steward must enrich is to select a number of categories and govern only the data that falls within those categories. Another is to only govern known pipelines of data (these are the pipelines consisting of the primary data a company deals with, such as daily sales numbers from a retail store) and to handle "ad hoc" pipeline data (engineers at times are asked to create *new* pipelines to address one-time or infrequent data analysis use cases) only as time allows or if absolutely necessary.

It's easy to see that these strategies (which we will discuss in more depth later) result in a sort of iceberg of data where enriched (*known*), curated data sits at the top, and below that is a mound of data that is, in the main, un-enriched and thus *unknown*. Companies don't know what this data is, which means they can't govern it, and ungoverned data is quite scary. It could be sensitive, it could be noncompliant, and there could be dire consequences if it happens to get leaked. This causes much fear and forces companies with this problem to use strategies to help mitigate their risk.

Not only do large companies deal with copious amounts of data, but they also tend to have a much larger workforce of data analysts, data scientists,

and data engineers—all of whom need access to data to do their jobs. More data, plus more people who need access to data, results in much more complicated (and often poorly managed) processes around access control. Access control is generally based on user role, which should determine the data (and only that data) that they need access to.

This strategy may seem simple, but it is a difficult process to implement for two main reasons. First, in order to know what data is critical to a particular user's role, data must be known. And we know from previous discussion in this book that the majority of data a company has is unknown. This results in not only the inability to govern (it's impossible to govern what you don't know you have) but also the inability to know what data is appropriate for whom. Data specialists still need to do their job, however, so (too much) access is often granted at the expense of risk. Companies try to offset this risk by creating a "culture of security," which puts the onus on the employee to do the right thing and not expose or misuse potentially sensitive information. Another issue with role-based access is that roles and their ensuing uses for data are not always black and white; the use case for data must also be considered. Depending on the company, a user in the same role may be able to use data for some use cases and not for others (an example being the use case we outlined in the retail section). As touched on in Chapter 2, and as will be covered in depth later, the addition of use case as a parameter during policy creation helps to mitigate this issue.

Like legacy companies, large companies often also deal with the issue of having many different storage systems, the majority of which are legacy. Large companies tend to be built up over time, and with time comes more data and more storage systems. We've already discussed how the siloed nature of different storage systems makes governance difficult. Different storage systems naturally create data silos, but large companies have another factor that results in even more complex "silos": acquisitions.

Large companies are sometimes built completely from within, but others become large (or larger) due to acquisitions. When companies acquire other, smaller companies, they also acquire all their data (and its underlying storage), which brings along a whole host of potential problems—primarily,

how the acquired company handled its data, as well as its overall governance process and approach. This includes how the company managed its data: its method of data classification, its enterprise dictionary (or lack thereof), its process and methods around access control, and its overall culture of privacy and security. For these reasons, many large companies find it nearly impossible to marry their central governance process with that of their acquisitions, which often results in acquired data sitting in storage and not being used for analytics.

The same multi-cloud reality compounds these problems at scale. A large enterprise's data repositories, operational applications, and AI models now routinely span several cloud ecosystems—some inherited through acquisition, some chosen deliberately to avoid lock-in—so that no single warehouse, and no single governance console, sees the whole picture. The response we increasingly see is a move away from both the rigid central warehouse and the siloed line of business toward a decentralized *data product model*, in which data is treated as a product with a clear owner, and ownership boundaries are mapped directly onto the engineering squads that build and run the pipelines. The squad that produces a dataset is accountable for its quality, its documentation, and its governance, which puts responsibility where the knowledge already lives. How this model resolves the silos it inherits is something we return to in the strategies discussed later in this chapter.

People and Process Together: Considerations, Issues, and Some Successful Strategies

We have now outlined the different people involved in various kinds of companies, as well as some specific processes and approaches utilized by the different company types.

There is obviously a synergy between people and process, of which there are some considerations and issues. We will review several of the issues

we've observed, along with outlining a few strategies we have seen in which the right people and process together have resulted in moving toward a successfully implemented data-governance strategy.

Considerations and Issues

This certainly is not an exhaustive list of all the potential issues with implementing a successful data-governance strategy; we are simply highlighting some of the top issues we've observed. We only briefly note the mitigation efforts to combat these issues; the latter half of this text will go into these in much more detail.

"Hats" versus "roles" and company structure

We previously discussed our intentional use of the terms *hats* versus *roles* and the impact that wearing many hats has on the data governance process. To expand on that idea further, an additional issue that arises with hats versus roles is that responsibility and accountability become unclear. When looking at the different kinds of approaches that different companies take to achieve governance, an underlying need is for actual people to take responsibility for the parts of that process. This is easy when it is clearly someone's *job* to conduct a piece of the process, but when the lines between what is and what is not within a person's purview are blurred, these fuzzy lines often result in inadequate work, miscommunication, and overall mismanagement. It's clear that a successful governance strategy will rely not simply on roles but on tasks, and on who is responsible or accountable for these tasks.

Tribal knowledge and subject matter experts (SMEs)

When talking with customers about their pain points with data governance, one of the things we hear over and over again is that they need tools to help their analysts find which datasets are "good," so that when an analyst is searching for data, they know that *this* dataset is of the best quality and is the most useful one for their use case. The companies state that this would help their analysts save time searching for the "right/best" dataset, as well

as assisting them in producing better analytics. Currently, through most companies, the way analysts know which datasets they should work with is by word of mouth, or "tribal knowledge." This is an obvious problem for companies because roles change, people move on, etc.

Companies request "crowdsourcing" functionality, such as allowing analysts to comment on or rank datasets to help give them a "usefulness" score for others to see when searching. This suggestion is not without merit but uncovers the larger problems of *findability* and *quality*. Companies are relying on *people* to know the usefulness of a dataset and to transfer that knowledge on to others, yet this strategy is fallible and difficult (if not impossible) to scale. This is where tools that lessen (or negate) the effort placed on a particular user or users aid in the process. A tool that can detect the most-used datasets and surface these first in a search, for example, can help minimize the reliance on tribal knowledge and SMEs.

An AI dataset-discovery engine addresses the same need without relying on people to remember. Rather than waiting for analysts to crowdsource a "usefulness" score, the engine ranks datasets by lineage, query frequency, freshness, and schema quality across clouds, and answers a plain-language request—"where is the authoritative customer-revenue table?"—with the best candidate and the evidence for choosing it. This does not abolish tribal knowledge so much as encode it, so that it survives the people who leave.

Definition of data

Regardless of their type, *all* companies want to be able to collect data that can be used to drive informed business decisions. They are certain that more data, and the analytics that could be run on that data, could result in key insights—insights that have the potential to skyrocket the success of their business. The problem, however, is that in order for data to be used, it must be *known*. It must be known what the letters or numbers or characters in a column of a table mean. And now it must *also* be known whether those numbers, letters, or characters represent information that is sensitive in nature and thus needs to be treated in a specific way. Data enrichment is key to "knowing" data, yet it is largely a manual process. It generally requires

actual people to look at each and every piece of data to determine what it is. As we discussed, this process is cumbersome on its own, and it becomes almost impossible when the extra complexity of disparate data storage systems and different data definitions and catalogs are considered. The "impossible" nature of this work in general means that it just never gets done; this leaves companies scrambling to use a few tools and to implement some half strategies to make up for it—and also hoping that educating people on how data *should* be treated and handled will somehow be enough.

Old access methods

Gone are the days of simple access controls—i.e., these users/roles get access and these users/roles do not. Historically, there were not many users or roles who even had the *need* to view or interact with data, which meant that only a small portion of employees in a company needed to be granted access in the first place. In today's data-driven businesses there is the potential for many, many users who may need to touch data in a variety of ways, as seen in the beginning of this chapter. Each hat has different types of tasks that it may need to do in relation to data that requires varying levels of access and security privilege.

We already discussed the problematic nature of unknown data; another layer to that is the implementation of access controls. There is a synergy between knowing which data even *needs* access restrictions (remember, data must be known in order for it to be governed) and knowing what those restrictions should be for what users. As discussed in Chapter 2, there are varying levels of access control, all the way from access to plain text to access to hashed or aggregated data.

A further complication around access is that of the *intent* of the user accessing data. There may be use cases for which access can and should be granted, and other use cases for which access should strictly be denied. A prime example (and one we've heard from more than one company) is a customer's shipping address. Imagine the customer just bought a new couch, and their address was recorded in order to fulfill shipment of that

couch. A user working to fulfill shipping orders should undoubtedly get access to this information.

Now let's say that the slipcovers for the couch this customer just bought go on sale, and the company would like to send a marketing flyer to the customer to let them know about this sale. It may be the case that the user in the company who handles shipping data also happens to handle marketing data (remember hats?). If the customer has opted *out* of promotional mail, the user in this example would not be able to send the marketing material *even though* they have access to that data. This means that access controls and policies need to be sensitive enough to account not only for a black-and-white "gets access/doesn't get access" rule for a particular user, but also for what *purpose* the user is using the data.

AI access methods

The limitations of access control models designed for human users are significantly magnified when applied to AI agents. Unlike human operators, AI agents lack static organizational roles to anchor traditional Role-Based Access Control (RBAC) frameworks. Their permissions rely entirely on transient intents and specific use cases. Crucially, these parameters can shift dramatically from one invocation to the next. Traditional binary permission models—granting or denying access based solely on static identity—cannot accommodate this operational fluidity.

Therefore, authorization mechanisms must transition to real-time, context-aware evaluations. The system must dynamically verify the requesting entity, the delegator, the explicit operational purpose, and the corresponding policy constraints prior to execution. This necessity for granular control applies down to the micro-level of individual agent requests. For instance, a system must permit an order-fulfillment agent to access a shipping address while strictly obfuscating that same data from a marketing agent acting for an opted-out customer, even if the foundational model possesses the technical capability to process both tasks.

While real-time context evaluation effectively resolves the authorization of an actor, it does not address the inherent trustworthiness of the subsequent

action. Mitigating this post-authorization risk requires a formalized governance matrix that categorizes system operations by their potential impact.

High-risk executions—such as bulk data deletion, the export of sensitive data columns, or modifications to Identity and Access Management (IAM) policies—mandate strict Human-in-the-Loop (HITL) protocols. Under HITL, operations remain suspended until explicit manual approval is granted. Conversely, routine, high-throughput automated tasks cannot sustain the latency of manual gating, nor is such friction necessary. These operations are better served by a Human-over-the-Loop (HOTL) architecture. This approach permits autonomous agent execution while administrators continuously audit logs, sample algorithmic decisions, and retain the overriding authority to intervene. Ultimately, this governance matrix functions as a proactive, codified policy, mapping specific operations to their corresponding oversight tiers long before an incident forces a reactive response.

Regulation compliance

Another struggle that companies have is around compliance with regulations. Some regulations, such as those in financial and healthcare industries, have been around for quite a while, and as we pointed out before, companies who deal with these kinds of data tend to have better governance strategies, for two main reasons: one, they've been dealing with these regulations for some time and have built in processes to address them, and two, these regulations are fairly established and don't change much.

The advent of a proliferation of data collection has brought about new regulations such as GDPR and CCPA that aim to protect *all* of a person's data, not just their most sensitive data (i.e., health or financial data). Companies in all types of industries, not just highly regulated ones, now must comply with these new regulations or face serious financial consequences. This is a difficult endeavor for companies that previously did not have regulations to comply with and thus perhaps did not address this during the setting up of their data infrastructure. As an example, one of the

main components of GDPR is the “right to be forgotten,” or the ability for a person to request that all their data collected by a company be deleted. If the company does not have its data set up to *find* all the permutations of a person’s individual data, it will struggle to be compliant. Thus, it can be seen how findability is important not only from the perspective of finding the right data to analyze but also from the perspective of finding the right data to delete.

In addition to GDPR and CCPA, you may have to account for AI-specific regulatory needs by using frameworks that go beyond privacy and into lineage. The old question was where a person’s data lives so it can be deleted. The new one is where the data used to train a model came from, and whether the company was entitled to use it that way. Lineage has to answer both. It must trace the ethical provenance of a training set as readily as the permutations of an individual’s records, proving not only that a deletion request can be honored but that a model’s inputs were lawfully sourced. Findability built solely around the right to be forgotten does not reach that far.

CASE STUDY: GAMING OF METRICS AND ITS EFFECT ON DATA GOVERNANCE

Be aware of the human element when you assign regulations and compliance. A relevant case study about how, when introducing metrics, you should factor in the human response to these metrics is in Washington, DC's school system. In 2009, Washington, DC introduced a new ranking system called IMPACT through which teachers were assessed and scored. The system, born out of the intention to promote "good" teachers and generally improve education—and, even further, to allow "low scoring" teachers to be dismissed—actually did not achieve the desired result.

The ranking system was based on an algorithm that took into account standardized test scores of students, with the underlying assumption that test scores for students with "good" teachers should show improvement year after year. However, the ranking system did not include social and familial circumstances, nor did it include any kind of feedback or training from a controlled set.

The result of implementing the system was that teachers were let go based only on the results of the standardized tests, and without regard to feedback from administrators. The immediate reaction was that teachers pivoted to focusing on how to pass the standardized tests rather than on the underlying principles of the subjects. This resulted in high scores and advancement for teachers whose students passed the tests, but it did not, sadly, promote excellence among the students.

In fact, in 2018 a city-commissioned investigation revealed that one in three students graduating in 2017 received a diploma despite missing classes. After being challenged, and after a detailed account of the usage of the algorithm and its usefulness had been carried out, the system was eventually "reconsidered" in 2019.

The process lessons here are that, although the motivation (improving teaching, promoting excellent teachers) should have been easily agreed

upon by both teachers' unions and the school district administrators, the implementation was lacking. A proper way to introduce a process is to first have a discussion with the affected people. Take feedback to heart and act on it. Run the process for the first time as a trial, with transparent results, open discussion, and a willingness to pivot if it isn't working as intended.

Processes and Strategies with Varying Success

The issues with people and processes are many, and we have observed some strategies that have been implemented with varying degrees of success. While we will explore some of these strategies here, the latter part of this book dives deeper into how these strategies (and others) can be implemented to achieve greater data governance success.

Data segregation within storage systems

We've reviewed some of the issues that arise from having multiple data-storage systems; however, some companies use multiple storage systems and even different storage "areas" within the same storage system in an advantageous and systematic way. Their process is to separate curated/known data from uncurated/unknown data. They may do this in a variety of ways, but we have heard two common, prevailing strategies.

The first strategy is to keep all uncurated data in an on-prem storage system and to push curated data that can be used for analytics to the cloud. The benefit that companies see in this strategy is that the "blast radius," or the potential for data to be leaked either by mistake or by a bad actor, is greatly diminished if only known, clean, and curated data is moved into a public cloud. To be clear, companies often indicate that it is not their total distrust of cloud security that is the problem (although that can play a role as well); it is their concern that *their own employees* may unwittingly leak data if it's in a public cloud versus an on-prem environment.

Figure 2-4 shows an example of data stored on-premises and in the cloud. As you can see, even though both storage systems have structured datasets,

only the cloud dataset has been enriched with metadata and thus has been treated with the appropriate governance controls (hashing in this case). The on-premises dataset, while it has the same information, hasn't been curated or enriched; we don't know just by looking at it that the numbers in the first column are credit card numbers or that the words in the second column are customer names. Once these columns have been curated and the appropriate metadata (credit card number, customer name, etc.) has been attached, we can attach governance controls, so that even if the data gets leaked, it will be the protected version and not plain text.

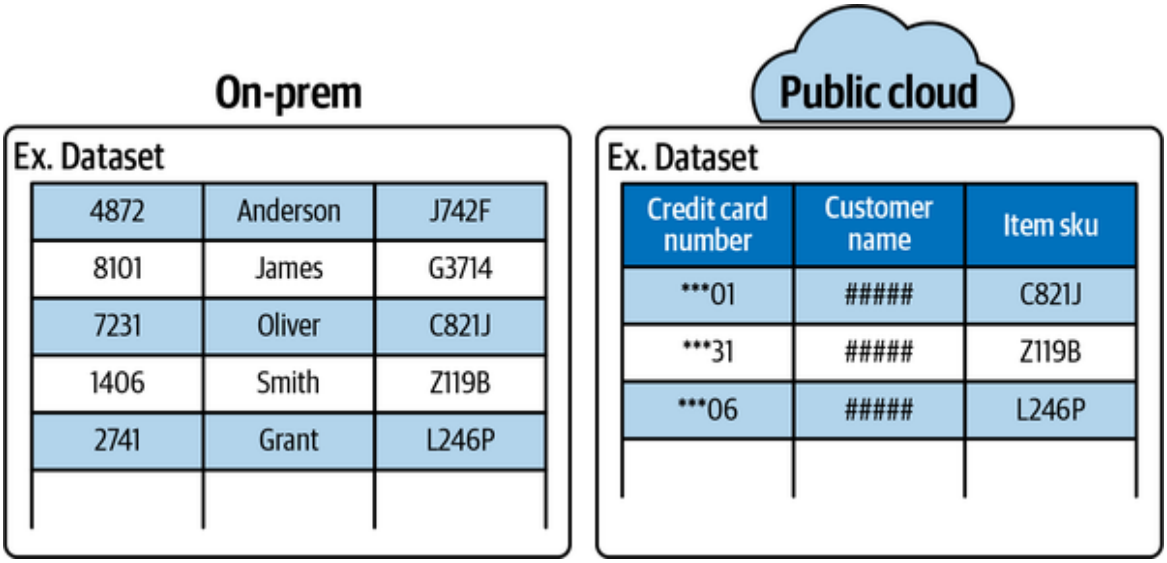


Figure 2-4. Enriched versus not enriched datasets on-premises and in the public cloud

As we've stated, an obvious benefit of this strategy is that if sensitive data resides only on-premises, then if it's leaked or incorrectly accessed, that could only have been done by someone within the company. On the other hand, if that same sensitive data is in a public cloud, it could potentially have been accessed by anyone if leaked or hacked into.

There are a few drawbacks to this strategy, however. One is that when data is segregated this way—some residing on premise and some residing in the cloud—cross-storage analytics are difficult if not impossible to complete. Since one of the main drivers of collecting so much data is being able to run powerful analytics, hindering this seems counterproductive. The other drawback is that segregation also requires upkeep of and attendance to these multiple storage systems and data pipelines, not to mention the creation,

maintenance, and enforcement of additional access controls, all of which are difficult to properly manage and stay on top of over time.

The second strategy is similar to the first in that there is a separation between curated and uncurated data, but this is done within the same cloud environment. Companies will create different layers or zones (as can be seen in Table 2-2) within their cloud environment and base access on these; the bottom, uncurated zone may be accessed by only a few users, while the uppermost zone, curated and cleaned (of sensitive data), may be accessed by any user.

Table 2-2. Table 3-2. A data lake within cloud storage with multiple tiers showing what kind of data resides in each and who has access

	Types of data	Access
Insights zone	Known, enriched, curated, and cleaned data. Data also has likely had governance controls such as encryption, hashing, redaction, etc. *Example: Well-labeled, structured datasets.*	Highest level of access. Most if not all data analysts/scientists and others in a user role.
Staging zone	More known and structured data. Data from multiple sources is likely to be joined here. This is also where data engineers prep data, cleanse it, and get it ready to drop into the insights zone.	More access. Mostly data engineers—those in charge of putting together datasets for analytics.
Raw zone	Any kind of data. Unstructured and uncurated. Could also include things such as videos, text files, etc. *Example: Video files, unstructured datasets*	Very restricted access. Likely a handful of people or just an admin.

Clearly the benefits and drawbacks of this strategy are nearly a reverse of those of the previous strategy. In this strategy, management and upkeep of the storage system, data pipelines, and policies are limited to one system, which makes it far simpler and more streamlined to stay on top of. Analytics are also largely easier to run because they can all be run within one central storage system, as opposed to trying to run them across multiple

systems, or dealing with the constant moving of data from one storage system to another for the purposes of analysis.

The raw zone was once a holding pen. Anything could be dropped there, structured or not, until someone got around to it. That casualness no longer survives the modern workload. Unstructured files, vector databases, and LLM training corpora all accumulate in this tier, and these are the assets most likely to hide sensitive information where column-level classification will never look: a face in a video frame, a name spoken in a call recording, a customer ID buried in a Markdown runbook. The raw zone has to be run as a managed repository now. It is no longer a place to defer the work.

Automated compliance gates make that tractable. Before a file reaches public cloud infrastructure, a gate transcribes and scans voice, detects faces and text in video, and parses free-form documentation. Anything sensitive is redacted or quarantined, and a record is kept of what was found. The point is to inspect at ingestion, while the blast radius is still small, rather than after the data has spread. As elsewhere, the machine enforces but does not decide; what counts as sensitive, and what must be blocked outright, stays with the governance team.

As we've noted, all data residing within a public cloud does carry the potential drawback of data leaking (whether intentionally or unintentionally) out onto the public internet—a cause for trepidation to be sure.

Data segregation and ownership by line of business

As we've mentioned more than a few times, data enrichment is a key challenge in successful data governance for many reasons, the primary ones being level of effort and lack of accountability/ownership.

One way we've observed companies handle this issue is by segregating their data by line of business. In this strategy, each line of business has dedicated people to do the work of governance on just that data. While this is often not actually delineated by role, each line of business has a deep knowledge of the kind of business it has and handles the ingress and egress

of data (pipelines), data enrichment, enforcement/management of access controls and governance policies, and data analysis. Depending on the company size and/or complexities of data within each line of business, these tasks may be handled by just one person, a handful of people, or a large team.

There are several reasons this process tends to be quite successful. The first is that the amount of data any given "team" must wrap its arms around is smaller. Instead of having to look at and manage a company's *entire* data story, the team needs to understand and work on only a portion of it. Not only does this result in less work, but it also allows for deeper knowledge of that data. Deeper knowledge of data has a multitude of advantages, a few being quicker data enrichment and the ability to run quicker, more robust analytics.

The second reason this process is successful is that there is clear, identifiable ownership and accountability for data: there is a specific person (or persons) to go to when something goes wrong or when something needs to change (such as the addition of a new data source or the implementation of a new compliance policy). When there is no clear accountability and responsibility for data, it's easy for that data to be lost or forgotten or worse —mismanaged.

In Figure 2-5, we've tried to show an example flow of different lines of business into a central repository. As you can see, retail store sales, marketing, online sales, and HR not only all feed into the central enterprise data repository in this example, but they are also *fed by* this repository.

To give you an idea of the different key hats and their tasks in each line of business, let's take marketing as an example. In this line of business in our example, we have the hats of data owner, data steward, and business analyst.

The data owner sets up and manages the pipelines in this line of business, along with managing requests for new pipelines and ingestion sources. They also perform the tasks of monitoring, troubleshooting, and fixing any

data quality issues that arise, as well as implementing any of the technical aspects of the company's governance policies and strategies.

The data steward is the *subject matter expert* (SME) in this line of business; knowing what data resides here, what it means, how it should be categorized/classed, and what data is sensitive and what isn't. They also serve as the point of contact between their line of business and the central governing body at their company for staying up to date on compliance and regulations, and they're responsible for ensuring that the data in their line of business is in compliance.

Finally, the business analyst is the expert on the business implications for the data in this line of business. They're also responsible for knowing how their data fits into the broader enterprise, and for communicating which data from their line of business should be used in enterprise analytics. In addition, they need to know what additional/new data will need to be collected for this particular line of business to help answer whatever the current or future business questions are.

From this example you can see how each hat has their role and tasks for just this one line of business, and how a breakdown of any of those tasks can result in a less than efficient flow/implementation of a governance program.

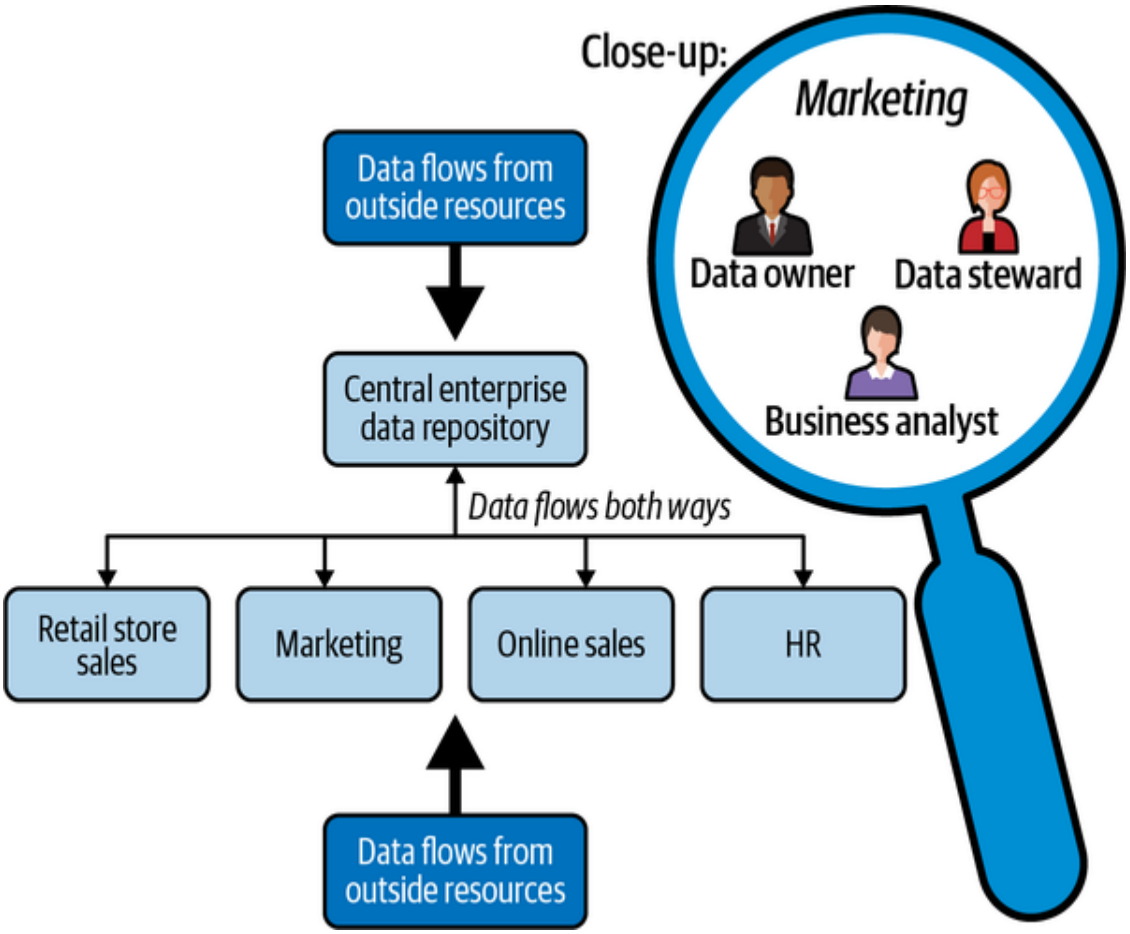


Figure 2-5. Flowchart of an enterprise with four lines of business and their data ingress/egress, as well as a close-up of one line of business, its key hats

---

This process, while successful, does come with some pitfalls. The primary one is that segregating data by line of business encourages the siloing of data and can, depending on how it's set up, inhibit cross-company analytics.

We recently worked with a large retail company that was dealing with this exact issue. This large US retailer (with over 350,000 employees), in an effort to better deal with the tremendous amount of data it collects, divided up its data by several lines of business. These included air shipping, ground shipping, retail store sales, and marketing. Having the data separated in this way greatly enabled the company to devote specific data analytics teams to

each line of business, whereby it could more quickly enrich its data, allowing it to apply governance policies to aid with CCPA compliance.

This strategy, however, created issues when the company began to want to run analytics *across* its different lines of business. To separate its data, the company created infrastructure and storage solutions with data pipelines that fed directly (and only) into one or another line of business. We won't go into too much depth on data pipelines here, but in short, it is a common practice to have data "land" in only one storage solution, because duplicating data and transferring it to additional storage areas is costly and difficult to maintain. Because specific data resided only in one storage area/silo, the company could not run analytics across its lines of business to see what patterns might emerge between, for example, air shipping and retail store sales (Figure 2-6).

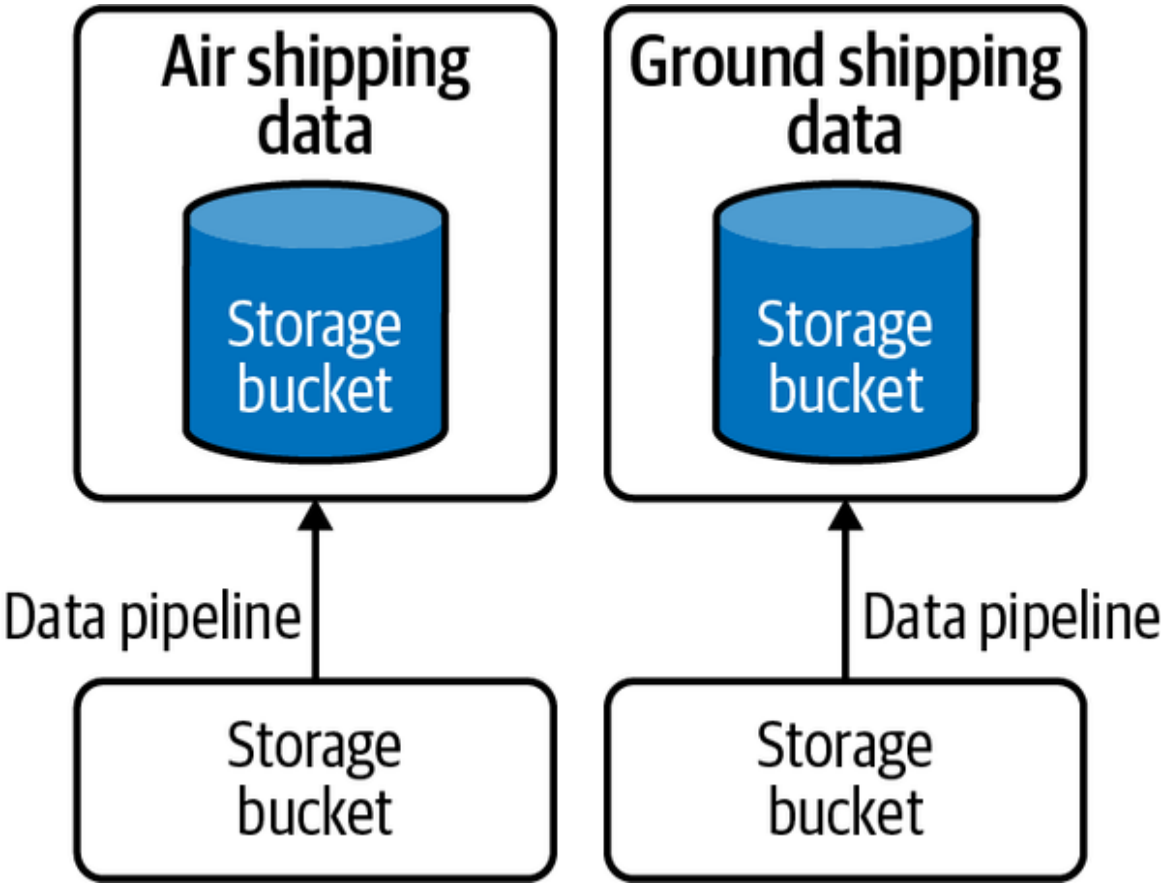


Figure 2-6. The preceding example company's two data silos: air shipping and ground shipping. Each silo has its own data pipeline and stores this data within its own bucket, meaning analytics can be run only within silos unless data from that pipeline is duplicated and piped into another silo.

This process of segregation by line of business to aid with accountability and responsibility is not a bad strategy, and while it's obvious there are pitfalls, there are also ways to make it more successful, which we will cover later in the book.

The decentralized successor to this strategy addresses the duplication pitfall head-on. Rather than physically copying a dataset out of one silo and piping it into another so that cross-line analytics can run—the costly, hard-to-maintain step that defeated the retailer above—federated governance operates over a *virtualized* data layer. Queries reach across the underlying stores in place, and governance policies, lineage, and access controls are enforced at that federated layer rather than re-implemented in each silo. Ownership still maps cleanly onto the squad that produces each data product, preserving the accountability that made line-of-business segregation work; what changes is that massive historical datasets no longer have to be migrated or replicated to be governed and analyzed together. The silo as an organizational boundary survives; the silo as a wall around the data does not.

Creation of "views" of datasets

A classic strategy employed by many companies is to create different *views* of datasets. As seen in Table 2-3, these views are really just different versions of the same dataset and/or table that have sensitive information sanitized or removed.

Table 2-3. Table 3-3. Three potentially different types of views of data: one plain text, one with sensitive data hashed, and another with sensitive data redacted

Plain text customer name	Hashed customer name	Redacted customer name
Anderson, Dan	Anderson, #####	********
Buchanan, Cynthia	Buchanan, #####	********
Drexel, Frieda	Drexel, #####	********
Harris, Javiar	Harris, #####	********

This strategy is classic (and works) because it allows analytics to easily be run, worry free, by virtually anyone on the "clean" view (the one with sensitive data either hashed or removed). It takes away the risk of access to and usage of sensitive data.

While this strategy works, it is problematic in the long run for several reasons. The first is that it takes quite a bit of effort and manpower to create these views. The clean view has to be manually created by someone who *does* have access to all of the data. They must go in and identify any and all columns, rows, or cells that contain sensitive data and decide how they should be treated: hashed, aggregated, completely removed, etc. They then must create an entirely *new* dataset/table with these treatments in place and make it available to be used for analytics.

The second issue is that new views constantly need to be created as fresher data comes in. This results not only in much time and effort being put into creating "fresh" views but also in a proliferation of datasets/tables that are difficult to manage. Once all these views are created (and re-created), it's hard to know which is the most fresh and should be used. Past

datasets/tables often can't be immediately deprecated, as there may be a need down the line for that specific data.

While views have some success and merit in aiding in access controls, we will propose and discuss some strategies we have seen that not only are easier to manage but scale better as a company collects more and more data.

To mitigate the operational overhead of continuous data duplication, organizations could transition to dynamic data masking (DDM). Rather than materializing a distinct, sanitized replica upon every data ingestion cycle, obfuscation is applied dynamically at query time against a centralized repository. The underlying data remains consolidated in a single location.

The specific representation exposed to an inquiring user or AI agent—whether plaintext, cryptographically hashed, or fully redacted—is computed on the fly based on centralized governance policies. Crucially, this unified policy framework extends across multi-cloud architectures, eliminating the redundancy of configuring localized access rules for each individual datastore. Programmatic redaction and hashing protocols effectively replace the manual maintenance of static views.

Because no secondary datasets are generated, the architecture completely eliminates the need to track, reconcile, or deprecate duplicate records. Consequently, the inherent risks of querying stale data are eradicated. Ultimately, disparate data treatments cease to be isolated physical tables; instead, they function as true logical views projecting securely from a single, strictly governed source.

A culture of privacy and security

The final process we'd like to discuss is that of creating a culture of privacy and security. While certainly every company and employee should respect data privacy and security, the ways in which this strategy is thought through and implemented are truly a special ingredient in creating not just a good data governance strategy but a successful one.

We have dedicated an entire chapter to the strategy of building a successful data culture (see Chapter 9), so more on that is to come; however, as a short

introduction to the concept, we will note that we have seen companies driven to really look at their data culture and work to implement a new (successful) one because of one (or all) of the following occurring within their organization:

- Their governance/data management tools are not working sufficiently (meaning the tools do not in and of themselves provide all of the governance "functionality" desired by a company).
- People are accessing data they should not be accessing and/or using data in ways that are not appropriate (whether intentionally or not).
- People are not following processes and procedures put into place (again, whether intentionally or not).
- The company knows that governance standards/data compliance are not being met and don't know what else to do, but they hope that educating people on doing the "right thing" will help.

To be sure, throughout this text we have already covered in great detail governance tools, the people involved, and the processes that are/can be followed. This is where the importance of putting all the pieces together can be seen—i.e., tools, people, and process. One or even two pieces are insufficient to achieve a *successful* governance strategy.

As evidenced by the four reasons we've just listed, even though a few tools, some people, and several processes are in place, some gaps may still remain.

This is where it must all come together—in a collective *data culture* that encompasses and embodies how the company thinks about and will execute its governance strategy. That encompasses what tools it will use, what people it will need, and the exact processes that will bring it all together.

Modern security cultures must also address emerging behavioral risks associated with algorithmic governance. As automated AI systems increasingly drive oversight mechanisms—such as quality scoring,

compliance verification, and anomaly detection—human operators inevitably decipher these evaluation criteria. Consequently, individuals may begin to optimize their outputs for the metric itself rather than the foundational objective it represents.

This operational drift represents a contemporary manifestation of Goodhart's Law: once a measure becomes a target, it ceases to be a valid measure. For example, an operator who calibrates data labeling merely to appease a classification model, or who structures workflows specifically to bypass algorithmic compliance triggers, effectively neutralizes the control mechanism while projecting a facade of adherence.

To counteract this vulnerability, organizational culture must systematically expose and disincentivize metric manipulation. Mitigation requires the deployment of independent, out-of-band audits capable of detecting behavioral patterns that the primary metrics cannot observe. Furthermore, leadership must establish a definitive, enforceable mandate prioritizing fundamental operational integrity over superficial scoring.

Summary

In this chapter, we have reviewed multiple unique considerations regarding the people and process of data governance for different kinds of companies. We've also covered some issues commonly faced by companies, as well as some strategies we've seen implemented with varying success.

From our discussion it should be clear that data governance is not simply an implementation of tools but that the overall process and consideration of the people involved—while it may vary slightly from company to company or industry to industry—is important and necessary for a successful data governance program.

The process for how to think about data, how it should be handled and classified from the beginning, how it continually needs to be (re)classified and (re)categorized, and who will do this work and be responsible for it—

coupled with the tools that enable these tasks to be done efficiently and effectively—is key, if not mandatory, for successful data governance.

Artificial intelligence now sits on both sides of this equation. It appears among the people as a category of AI and agentic hats: the annotator who teaches models from raw data, the on-behalf-of agent that inherits a human's scope, and the autonomous agent that holds its own credentials and requires its own named owner. It appears among the tools as well. Generative copilots take a first pass at the soul-sucking work of classification, natural-language interfaces let business users answer their own questions, and dataset-discovery engines encode the tribal knowledge that would otherwise walk out the door when an employee leaves. In each case the machine proposes and a human still decides.

The process must stretch to match. Data routinely spans multiple clouds, which pushes companies away from the rigid central warehouse and the siloed line of business toward a decentralized data product model, federated governance over virtualized layers, and dynamic masking in place of hand-built views. The raw zone becomes a managed home for unstructured files and training corpora, guarded by automated compliance gates at ingestion. Access control moves from binary, role-based rules to real-time, context-aware evaluation, with an explicit matrix deciding when a human must approve an action (human-in-the-loop) and when it is enough to audit one after the fact (human-over-the-loop). The regulatory frontier moves too: lineage must establish not only where a person's data lives but where a model's training data came from. Tools, people, and process must come together—now with the added discipline of governing the AI that has joined them.

Chapter 3. Data Governance Over a Data Lifecycle

> **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. Please note that the GitHub repo will be made active later on.
>
> If you'd like to be actively involved in reviewing and commenting on this draft, please reach out to the editor at *ccollins@oreilly.com*.

In previous chapters, we introduced governance, what it means, and the tools and processes that make governance a reality, as well as the people and process aspects of governance. This chapter will bring together those concepts and provide a data life cycle approach to operationalize governance within your organization.

In this chapter you will learn about:

- Data life cycle and its phases across operations and analytics
- Governance best practices along each life cycle phase
- How the rise of AI systems further reinforces the need for governance and expand the types of data that require governance including derived data such as embeddings

Before getting into the detailed aspects of governance, it's important to center our understanding on data life cycle management and what it means

for governance.

You will learn about a data life cycle, the different phases of a data life cycle, data life cycle management, applying data governance over a data life cycle, crafting a data governance policy, best practices along each life cycle phase, applicable examples, and considerations for implementing governance. For some, this chapter will validate what you already know; for others, it will help you ponder, plant seeds, and consider how these learnings can be applied within your organization. This chapter will introduce and address a lot of concepts that will help you get started on the journey to making governance a reality. Before getting into the detailed aspects of governance, it's important to center our understanding on data life cycle management and what it means for governance.

What Is a Data Life Cycle?

Data life cycle is the order of stages data goes through from its initial generation or capture to its eventual archival or deletion at the end of its useful life.

It's important to point out that this definition tries to capture the essence of what happens to a piece of data; however, not all data goes through each phase, and these phases are simply logical dependencies and not actual data flows. As we learned in Chapter 2, data often doesn't physically move as it traverses its life cycle.

Organizations work with 2 types of data - unstructured and structured. Some organizations will also reference a semi-structured as a type of data, for simplicity, semi-structured can be treated as unstructured data:

Unstructured data

> Data that does not have a predefined data model or is not organized in a predefined way: free-form content such as text documents, email, images, audio, video, and web pages, which doesn't fit natively into relational rows and columns.

Structured data

> Data that conforms to a predefined data model, typically organized as records in rows and columns where each field has a defined data type, making it efficiently describable in a relational model and queryable with SQL. Records are conceptualized as the rows in a table where data elements are in the cells, and the form was traditionally captured as comma-separated values or a row in a relational database.

With the rapid growth of AI and Agents operating on our behalf, both types of data are used together. For example an agentic system processing insurance claims will use a RAG application based on the corpus of policy documents (unstructured) and payment & beneficiary data in a customer database (structured) to validate the claim.

Both structured and unstructured data is used for operations and analytics. Both operational and analytical data have their own life cycles driven by their purpose.

Type of Data	Operational	Analytics
Structured	Frequently accessed data stored in databases used for transactional processing. Also known as **OLTP** (online transactional processing).	Data used for decision making, generally accessed less frequently, often aggregated and stored in databases designed for analytic workloads such as Databricks and Snowflake. Also known as **OLAP** (online analytic processing).
Unstructured	Data generated or referenced for operational use cases such as customer service, financial transactions and health care. These are typically stored on collaboration platforms such as google drive, sharepoint and slack.	Data referenced for analytical insights and mining such as context graphs, training & eval sets.

Traditionally, when data does physically move - data flows from operational to analytic systems but with the rise of ML and advanced analytic processing, data can flow back and forth. For example, a CRM (Customer Relationship Management) application can send client signals to a ML process that in turn sends back an updated client score creating a cycle. The movement of analytic data back to an operational system is often referred to as reverse ETL.

Proper oversight of data throughout its life cycle is essential for optimizing its usefulness and minimizing the potential for errors. Data governance is at

the core of making data work for businesses. Defining this process end-to-end across the data life cycle is needed to operationalize data governance and make it a reality. And because each phase has distinct governance needs, this ultimately helps the mission of data governance.

Phases of a Data Life Cycle

As mentioned earlier, you will see a data life cycle represented in many different ways, and there's no right or wrong answer. Whichever framework you choose to use for your organization has to be the one guiding the processes and procedures you put in place. Each phase of the data life cycle as shown in Figure 3-1 has distinct characteristics. In this section, we will go through each phase of the life cycle as we define it, delve into what each phase means, and walk through the implications for each phase as you think about governance.

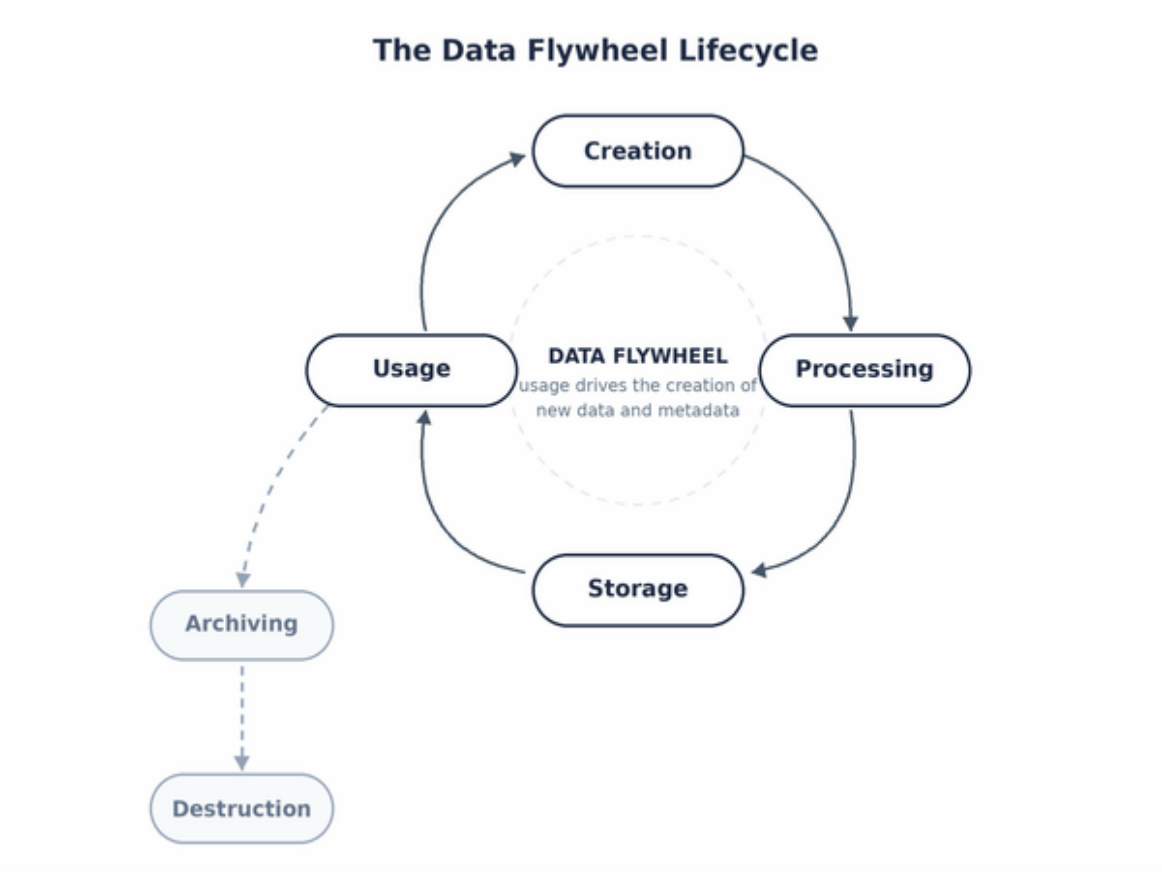


Figure 3-1. Phases of a data life cycle

Data Creation

The first phase of the data life cycle is the creation or capture of data. Data is generated from multiple sources, in different formats such as structured or unstructured data, and in different frequencies. For operations, data is often keyed in by internal users or external customers, generated by systems such as IoT devices and contract systems and as agentic systems become more prevalent, as a result of Agentic actions. Analytic data is created either by physically moving operational data into analytic stores such as lakehouses and warehouses via ETL processes or captured from external sources such as 3rd party enrichments. Derived data such as embeddings, aggregates, or synthetic data, which is used for both analytics and operationals can also be created and will progress through the same data life cycle.

Metadata—data about data such as provenance, usage guidelines and privacy tags —can also be created and captured in the creation phase and will closely follow the life cycle of the data it describes. Metadata management and use is critical for a successful governance program. Depending on the method data entered the enterprise will influence what type of metadata is generated.

You will notice *data creation* and *data capture* used interchangeably, mostly because of the source of data. When new data is created, that is referred to as data creation, and when existing data is funneled into a system, it is referred to as data capture.

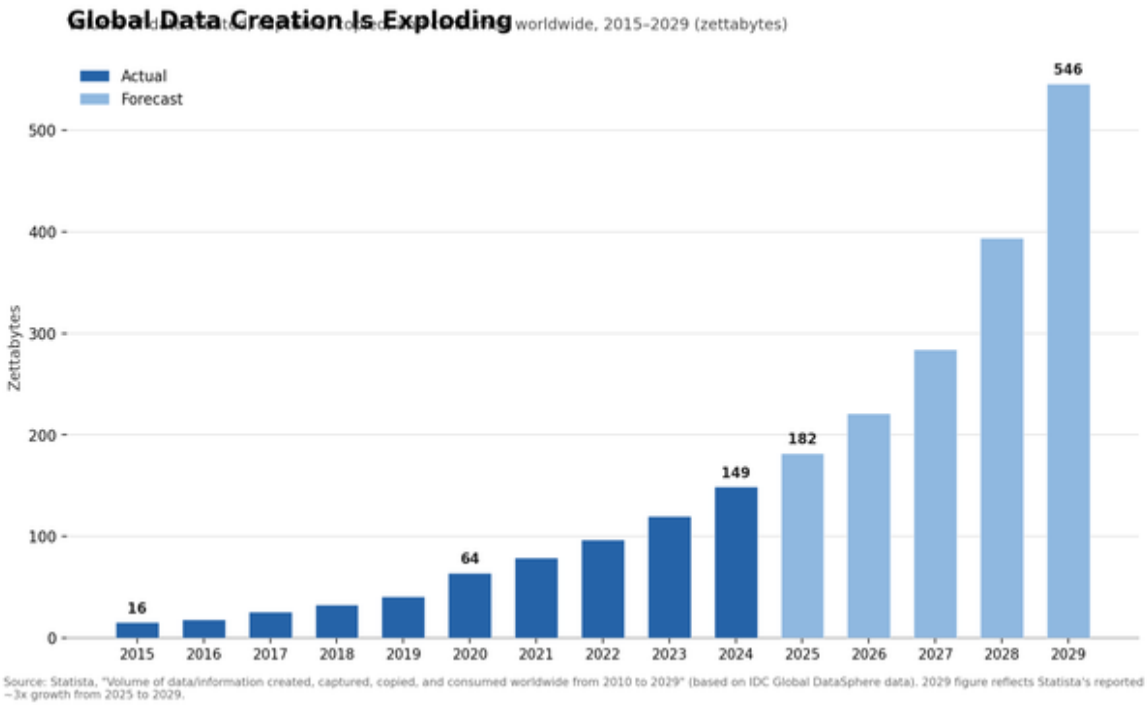


Figure 3-2. Global data creation year over year

As you can see in Figure 3-2, the amount of data created is on an exponential curve and we're just starting to enter the steep growth. With the expected exponential growth of agentic systems, the above chart is a conservative view. Today, data creation is limited by human to human and human to system interactions. In the very near future, agent to agent interactions and the associated data creation will exceed today's interactions by several orders of magnitude. Later in the chapter, we will look at how to think about governance during this phase, and we will call out the different tools you should think about when designing your governance strategy.

Data Processing

Operational Data Processing

Operational data must often be optimized for rapid reads and writes and therefore processing is limited to ensuring rapid transactions. Data is often mediated via REST APIs, event streams on frameworks such as Kafka and with the growth of AI systems and evolving standards, data is mediated via open standards designed for Agentic communication such as MCP (Model

Context Protocal) which enables LLM to resource communication & A2A (Agent to Agent) which enables agents to communicate with each other. Data processing happens on a continuous basis.

Governance controls referenced in Chapter 2 such as IAM and OBO for agents come into play during this step in the data lift cycle and are managed by the roles defined in Chapter 3.

Creating and capturing metadata during this stage is crucial for effectively managing all the other steps in the life cycle. Producing trace logs will ensure lineage graphs (Chapter 2) will show the full provenance of a data asset, applying descriptions to tables and columns in the OLTP databases will ensure consistent definitions are used everywhere, applying audit metadata when data is processed via agents will enable reconciliation between an agent's chain of reasoning and the end result of a data process.

Analytic Data Processing

Analytic data often requires additional processing, often in the form of a batch process in order to make it ready for analytics and ML. This preparation goes through a discrete set of steps - we'll use the medallion framework to define those steps in this book but there are others and although frameworks may differ, the approach is the same - data lands in a raw zone, data is prepared and data is curated for use.

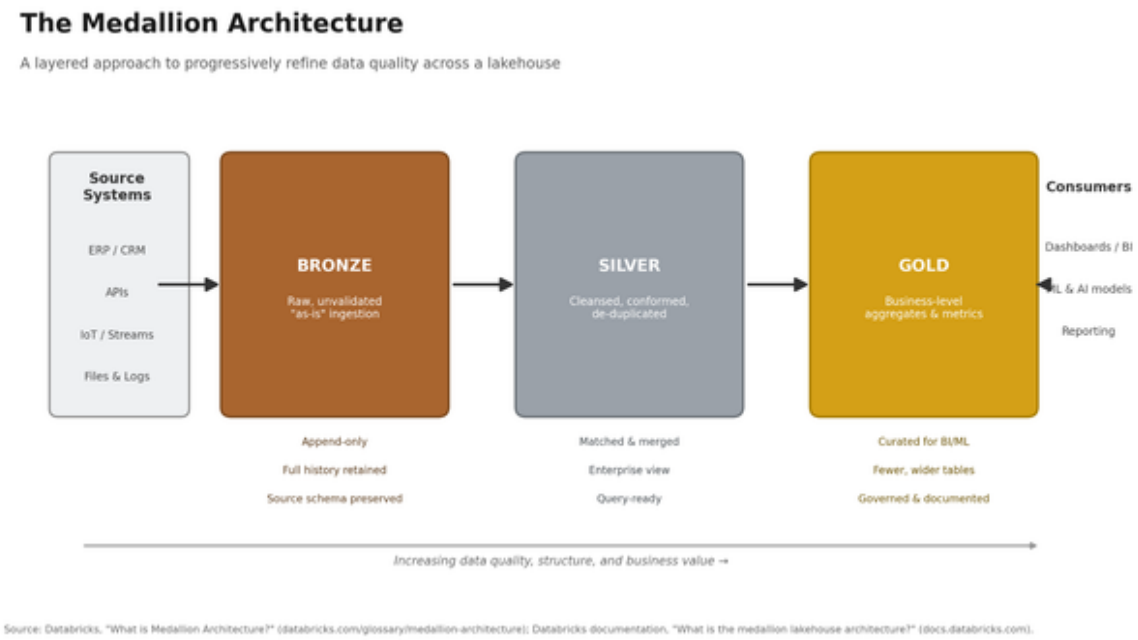


Figure 3-3. Caption to come

For a governance program to be effective, it's crucial to capture and manage metadata during the processing step for analytics.

Definition: Metadata - data about the data - this can encompass a wide variety of topics and is shaped by organizations needs. Subjects include:

- Column and table level descriptions - plain language descriptions for what the table and columns represent. For unstructured documentation, this could be a document description.
- Usage Restrictions - especially for 3rd party data - define use cases are permissible for the data (e.g. can only be used for marketing research but not for activation and audience generation)
- Business domain - to what business domain does this data belong to
- Data Classifications - both industry standards

Data lands in the bronze area or the raw zone - it lands "as is." (i.e., no processing has been applied). Generally, this is some type of cloud storage such as AWS's S3, Google Cloud's GCS or Azure data lake storage. The

critical governance implication is that the landing area is highly restricted, data is not meant for general consumption and as mentioned in Chapter 1, managing this zone is becoming more important with the growth in variety of objects that can live here. Leveraging Open Table Formats such as Apache Iceberg, Delta Lake or Hudi, also introduced in Chapter 2, enable you to store your data in an interoperable format, enabling teams to expose and process the data on a variety of platforms such as Snowflake, Databricks, BigQuery, Redshift and Microsoft Fabric. Open table formats also, crucially, allow you to store metadata with your data. This feature enables teams to simplify some of their governance processes by having a single source of truth for governance attributes.

We'll discuss data quality procedures at each stage in the next chapter.

> **NOTE**
>
> Packaging your metadata with your data ensures you can more readily manage your data through its lifecycle and ensure that any form of processing will respect governance rules ie. data classification and usage restrictions and greatly simplify understanding the semantics of the data.

Once it's landed, moving the data to silver can begin. This can entail physically moving it or placing it behind a virtual layer. This consists of applying metadata such as table and column level descriptions,, removing duplicates, adding audit columns and standardization i.e. converting to enterprise conforming formats and schemas. Teams will also capture profile metrics in the silver layer such as row counts, uniqueness, frequency distributions and null rates. The types of tests will depend on the nature of the data and downstream consumption needs. As mentioned earlier, tools such as Anomalo and Monte Carlo automate this, tools such as dbt offer low code ways of implementing these checks. Many organizations adopt hybrid approaches, leveraging data quality tools and custom tests in combination to create a full picture. These DQ metrics, which we'll dive into more deeply later in the chapter, along with the other metadata that's been captured are crucial metadata attributes that will form the context

graph, helping agents understand your data, improving accuracy and efficiency.

At this point, the data is ready for general consumption especially exploratory data analysis (EDA) but it may not be optimized for analytical, ML or Agnetic use. Data teams should be encouraged to build automation to create silver quality data as quickly as possible.

The transition from Silver to Gold is where you prepare the data for Analytics and ML usecases. Your Data & Analytics Engineers will shape the data, leveraging common patterns such as Star Schemas, Data Vaults and other dimensional modeling. As mentioned earlier, as you build these layers, ensuring that metadata is created and managed effectively is crucial for a scalable governance program.

In summary, some of the governance implications that you will come across are data lineage, data quality, and data classification. To make governance a reality, ensuring you treat metadata with the importance as the data itself, starting to capture it as early as possible

Data Storage

The third phase in the data life cycle is *data storage*, where both data and metadata are stored on storage systems and devices with the appropriate levels of protection. In any mid to large sized data environment, this will consist of a variety of storage formats and mediums. For operations that's usually OLTP databases such as Postgres and MySql and low latency key-value stores such as Google's bigtable and AWS's DynamoDB . For Analytics, that's Lakehouses based on open table formats and for Agentic use cases, that's Graph Databases for ontologies, and Vector databases for embeddings. As data platforms mature, these lines are increasingly blurred - many analytic and operational data bases now support vectors and many data platforms are blurring the lines between analytic and operational workloads.

Data Usage

The *data usage* phase is important to understanding how data is consumed within an organization to support the organization's objectives and operations. In this phase, data becomes truly useful and empowers the organization to make informed business decisions, drive workflows and derive insights. In this phase, humans and increasingly, agents will interact with data. The culmination of governance policies come together to ensure that those interactions will be trustworthy and secure.

Because data is consumed by multiple internal and external stakeholders and processes during this phase, proper access management and audits are key. In addition, there might be regulatory or contractual constraints on how data may actually be used, and part of the role of data governance is to ensure that these constraints are observed accordingly.

Data Archiving

In the *data archiving* phase, data and potentially metadata is removed from all active production environments. This can include moving data to lower tier storage such as Glacier on AWS and Archive on GCP & Azure, or offloading it from operational databases. The major cloud providers have services that can manage archival processes such as Intelligent tiering on AWS, smart tier on Azure and Autoclass on GCP. Archived data is no longer processed, used, or published, but is stored in case it is needed again in an active production environment.

A data governance plan should focus on archival automation that balances compliance requirements with cost considerations and updating metadata and context so that data consumers are aware that data has been archived. The plan should also consider what to do with derived data. For example, if a set of documents are archived, should their respective embeddings also be archived.

Data Destruction

In this final phase, data is destroyed. *Data destruction*, or *purging*, refers to the removal of every copy of data from an organization, typically done from

an archive storage location. Even if you wanted to save all your data forever, it's just not feasible. It's very expensive to store data that is not in use, and compliance issues create the need to get rid of data you no longer need. The primary challenge of this phase is ensuring that all the data is properly destroyed and at the right time.

Before destroying any data, it is critical to confirm whether there are any policies in place that would require you to retain the data for a certain period of time. Coming up with the right timeline for this cycle means understanding state and federal regulations, industry standards, and governance policies to ensure that the right steps are taken. You will also need to prove that the purge has been done properly, which ensures that data doesn't consume more resources than necessary at the end of its useful life.

You should now have a solid understanding about the different phases of a data life cycle and what some of the governance implications are. As stated previously, these phases are logical dependencies and not necessarily actual data flows.

We're sure you've heard the phrase "Rome was not built in a day," but that's really what this data life cycle is trying to do. Applying data governance in an organization is a daunting task and can be very overwhelming. However, if you think about your data within these logical data life cycle phases, implementing governance can be a task that can be broken down into each phase and then thought through and implemented accordingly.

Data Management Plan

A data management plan (DMP) defines how data will be managed, described, and stored. In addition, it defines standards you will use and how data will be handled and protected throughout its life cycle. You will primarily see data management plans required to drive research projects within institutions often tied to medical research, but the concepts of the process are fundamental to implementing governance especially with expected regulations tied to AI. The EU's flagship AI regulation is, in

operational substance, a data governance regulation as it requires dataset quality, bias examination, lineage, documented provenance, semantic clarity for downstream consumers, and accountable human oversight. Because of this, it's worth us doing a deep dive into them and seeing how these could be applied to implement governance within an organization.

With governance, you will quickly realize that there is no lack of templates and frameworks—see, for example, the DMPTool from Massachusetts Institute of Technology. You simply need to pick a plan or framework that works for your project and organization and march ahead; there's not one right or wrong way to do it. If you choose to use a data management plan, here is some quick guidance to get you started. The concepts here are much more fundamental than the template or framework, so if you were able to capture these in a document, then you'd be ahead of the curve.

Guidance 1: Identify the data to be captured or collected and extrapolate

Data volume is important to helping you determine infrastructure costs and people time. You need to know how much data you're expecting and the types of data you will be collecting:

Types

Outline the various types of data you will be collecting. Are they structured or unstructured? This will help determine the right types of infrastructure to use although most modern platforms can handle both as 1st class citizens.

Sources

Where is the data coming from? Are there restrictions on how this data can be used or manipulated? What are those rules? All of these need to be documented and stored with the data

Volume

This is difficult to predict, especially with the exponential growth of data along with the growth of derived data such as embeddings and context graphs counter-balanced against evermore powerful virtual layers and zero copy options that can dramatically reduce the amount of derived data you actually need to store. However, creating a budget for expected amount of data created based on historical trends and a multiplier to account for derived data such as creating AI and Analytics ready data along with the ability to track both will enable an organization to effectively project not only storage cost but also the size of the estate they are obligated to govern along with appropriate staffing levels.

Guidance 2: Define how the data will be organized

Using inputs from the data collection exercise, organizations can determine the type of tooling to effectively manage the data estate. This can include traditional OLTP databases, Lakehouses, Graph databases and other types of built for use data processing platforms. Along with data processing platforms, organizations should also consider discovery and monitoring tools such as Data Catalogs and Data Quality tools. These platforms will store and organize an organization's metadata and certain types of derived data such as business context. This will help organizations map governance policies to practical implementations.

Guidance 3: Document a data storage and preservation strategy

Disasters happen, and ensuring that you've adequately prepared for one is very important. How long will a piece of data be accessible, and by whom? How will the data be stored and protected over its life? As we mentioned previously, data purging needs to happen according to the rules set forth. In addition, understanding what your systems' backup and retention policies are, is important.

Guidance 4: Define data policies

It's important to document how data will be managed and shared. Identify the licensing and sharing agreements that pertain to the data you're collecting. Are there restrictions that the organization should adhere to? What are the legal and ethical restrictions on access to and use of sensitive data, for example? Regulations like GDPR and CCPA can easily get confusing and can even become contradictory. In this step, ensure that all the applicable data policies are captured accordingly and the appropriate tooling exists to be able to implement these policies.

Guidance 5: Define roles and responsibilities

Chapter 3 defined roles and responsibilities. With those roles in mind, determine which are the right ones for your organization and what each one means for you. Which teams will be responsible for metadata management and data discovery? Who will ensure governance policies are followed all the way? Will the organization adopt a centralized or federated governance model or something in between?

A DMP should provide your organization with a roadmap that will guide others and explain how data will be treated throughout its life cycle. Think of this as a living document that evolves with your organization as new datasets are captured, and as new laws and regulations are enacted.

Applying Governance over the Data Life Cycle

We've gone through fundamental concepts thus far; now let's bring everything together and look at how you can apply governance over the data life cycle. Governance needs to bring together people, processes, and technology to govern data throughout its life cycle. In Chapter 2, we outlined a robust set of tools to make governance a reality, and Chapter 3 focused on the people and process side of things. It's important to point out that implementing governance is complicated; there's no easy way to simply apply everything and consider the job done. Most technologies need

to be stitched together, and as you can imagine, they're all coming from different vendors with different implementations. You would need to integrate the best-in-class suite of products and services to make things work. Another option is to purchase a fully integrated data platform or governance platform. This is not a trivial task.

Data Governance Framework

Frameworks help you visualize the plan, and there are several frameworks that can help you think about governance across the data life cycle. Figure 3-4 is one such framework in which we highlight all the concepts from Chapter 2, overlaid with the concepts we've discussed in this chapter.

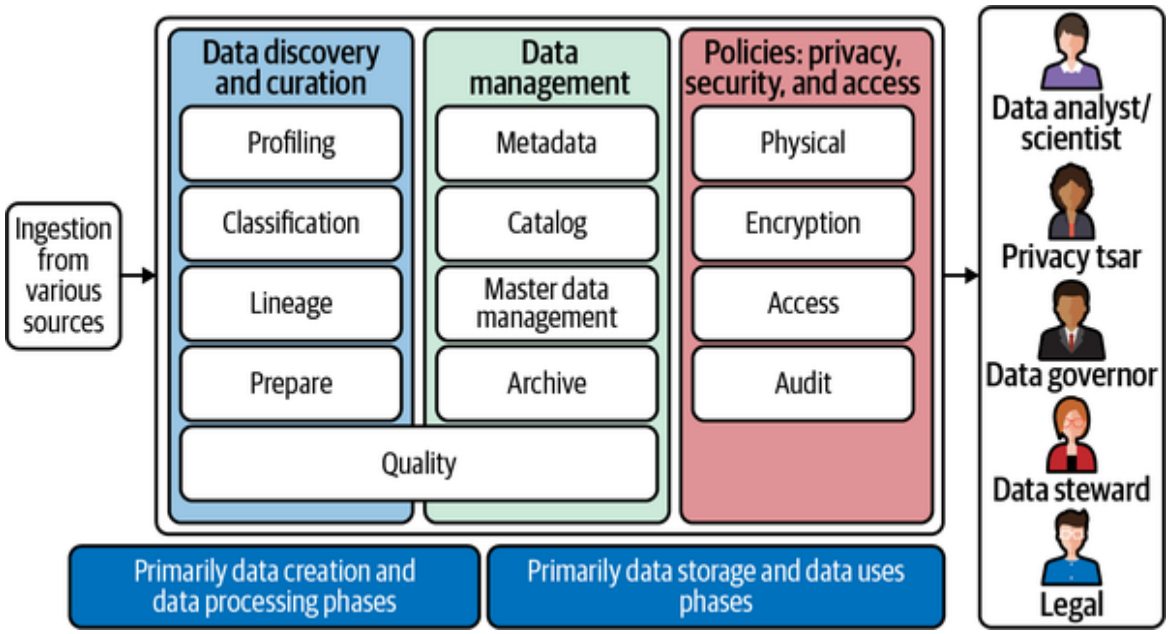


Figure 3-4. Governance over a data life cycle

This framework oversimplifies things to make them easier to understand; it assumes things are linear, from left to right, which is usually not the case. When data is ingested from various sources on the left, this is simply at the point of data creation or capture. That data is then processed and stored, and then it is consumed by the different stakeholders, including data analysts, data engineers, data stewards, and so on.

Data archiving and data destruction are not reflected in this framework because those take place beyond the point when data is used. As we previously outlined, during archiving, data is removed from all active production environments. It is no longer processed, used, or published but is stored in case it is needed again in the future. Destruction is when data comes to the end of its life and is removed according to guidelines and procedures that have been set forth.

We want to reiterate that Figure 3-4. provides a logical representation of the phases a piece of data goes through, from left to right, and not necessarily the actual step-by-step flow of the data. There's a lot of back and forth that happens between each phase, and not all pieces of data go through each one of these phases.

Frameworks are good at providing a holistic view of things. However, they are not the be-all, end-all. Make sure whichever framework you select works for your organization and your data.

> **NOTE**
>
> We've mentioned it already, but would like to emphasize again the idea of selecting a framework that works for your organization. This can include considerations around the kind of data you collect or work with, as well as what kind of personnel you have dedicated to your data governance efforts. One thing we challenge you to consider is how to take what you have and fit enough framework around it. Take these ideas as laid out (noting that not each step is required or even necessary) and layer on what you have to work with currently as a place to start. More pieces (and people, for that matter) can be added later, but if you focus on at least laying the groundwork—the foundation—you will be in a much better position if or when you *do* have more pieces to add to your framework.

Data Governance in Practice

OpenStreetMap (OSM) was created by Steve Coast in the UK in 2004 and was inspired by the success of Wikipedia. It is open source, which means it is created by people like you and is free to use under an open license. It was a response to the proliferation of siloed, proprietary international geographical data sources and dozens of mapping software products that

didn't talk to each other. OSM has grown significantly to over two million contributors, and what's amazing is that it works. In fact, it works well enough to be the trusted source of data for a number of Fortune 500 companies, including other small and medium-sized businesses. With so many contributors, OSM is successful because it was able to establish data standards early in the process and ensured contributors adhered to them. As you can imagine, a crowdsourced mapping system without a way to standardize contributor data could go wrong very quickly. Defining governance standards can bring value to your organization and provide trusted data for your users.

How Data Moves Through a Data Platform

Here's an example scenario of how data could move through a data platform with the framework in Figure 3-4. Let's say that a business wants to ingest data onto a cloud-data platform, like Google Cloud, AWS, or Azure, and share it with data analysts. This data may include sensitive elements such as US social security numbers, phone numbers, and email addresses. Here are the different pieces it might go through:

- Business configures an ingestion data pipeline using a batch or streaming service into the Bronze Layer
- Metadata is captured and generated:
    - Data is then scanned and classified for sensitive information such as PII and is tagged with PII tags/labels.
    - Some data may be redacted, obfuscated, or anonymized/de-identified. Metadata describing these changes is generated
    - Aspects of data quality can be accessed—that is, are there any missing values, are primary keys in the right format, etc.
    - Start to capture data provenance information for lineage.

  - Apply any descriptive details such as table and column descriptions and updates to any context graphs and skills files to ensure the data is discoverable by human and virtual analysts.

- Data is physically moved or virtually exposed in the Silver Layer and potentially transformed and joined with other data in the Gold Layer
- As data moves between the different services along the life cycle, it is encrypted in transit.
- Once ingestion and processing are complete, the data will need to be stored in a data warehouse and/or a data lake, where it is encrypted at rest. Backup and recovery processes need to be employed as well, in case of a disaster.
- Audit trails need to be captured throughout this data life cycle and made visible as needed. Audits allow you to check the effectiveness of controls in order to quickly mitigate threats and evaluate overall security health.
- Throughout this process, it is important to ensure that the right people and services have access and permissions to the right data across the data platform using a robust identity and access management (IAM) solution.
- You need to be able to run analytics and visualize the results for use. In addition to access management, additional privacy, de-identification, and anonymization tools may be employed.
- Once this data is no longer needed in a production environment, it is archived for a determined period of time to maintain compliance.
- At the end of its useful life, it is completely removed from the data platform and destroyed.

CASE STUDY: AIRCANADA'S CHATBOT AND THE BEREAVEMENT FARE

In February 2024, the British Columbia Civil Resolution Tribunal held Air Canada liable after its customer service chatbot gave a passenger inaccurate information about the airline's own refund policy.

In November 2022, Jake Moffatt asked the Air Canada chatbot which documents he needed to qualify for a bereavement fare, and whether the discount could be applied after the fact. The chatbot told him to book at full price and submit an online form within ninety days of the ticket being issued. Moffatt booked travel to and from Toronto to attend the funeral of a family member. When he applied for the refund, Air Canada denied it and pointed him to the bereavement section of its website, which stated that the policy did not apply to travel already completed.

Both statements were live on Air Canada's website at the same time. One was current, the other should have been retired, furthermore, the tribunal found that the chatbot was part of Air Canada's website and that the airline was responsible for all the information it published, no matter which system delivered it.

Although this case had a small financial penalty, the precedent is what matters. Air Canada had a lifecycle and a context problem: a superseded document stayed in circulation, and a system that could not tell current policy from retired policy handed it to a customer at the moment he was making a decision.

Every generative AI deployment inherits this risk, and in this case it comes from the end of the lifecycle. Governance programs generally invest heavily in ingestion, classification, and access control, and not enough in data retirement and updating metadata and context as a result of archival and deletion. Governance programs should invest consistently across the entire data lifecycle.

Operationalizing Data Governance

It's one thing to have a plan, but it's something else to ensure that plan works for your organization. NASA learned things the hard way. In September 1999, after almost 10 months of travel to Mars, the $125-million Mars Climate Orbiter lost communication and then burned and broke into pieces a mere 37 miles away from the planet's surface. The analysis found out that, while NASA had used the metric system, one of its partners had used the International System of Units (SI). This inconsistency was not discovered until it was time to land the orbiter, leading to a complete loss of the satellite. This of course was crushing to the team. After this incident, proper checks and balances were implemented to ensure that something similar did not happen again.

Bringing things together so that issues such as the one NASA experienced are caught early and rectified before a disaster happens starts with the creation of a data governance policy. A data governance policy is a living, breathing document that provides a set of rules, policies, and guidance for safeguarding an organization's data assets.

What Is a Data Governance Policy?

A data governance policy is a documented set of guidelines for ensuring that an organization's data and information assets are managed consistently and used properly. *A data governance policy is essential in order to implement governance*. The guidelines will include individual policies for data quality, access, security, privacy, and usage, which are paramount for managing data across its life cycle. In addition, data governance policies center on establishing roles and responsibilities for data that include access, disposal, storage, backup, and protection, which should all be familiar concepts. This document helps to bring everything together toward a common goal.

The data governance policy is usually created by a data governance committee or data governance council, which is made up of business, legal, compliance and technology executives and other data owners. This policy

document defines a clear data governance structure for the executive team, managers, and line workers to follow in their daily operations.

To get started operationalizing governance, a data governance charter template could be useful. Figure 3-5 shows an example template that could help you socialize your ideas across the organization and get the conversation started. Information in this template will funnel directly into your data governance policy.

Use the data governance charter template to kick off the conversation and get your team assembled. Once it has bought into your vision, mission, and goals, that is the team that will help you create and define your governance policy.

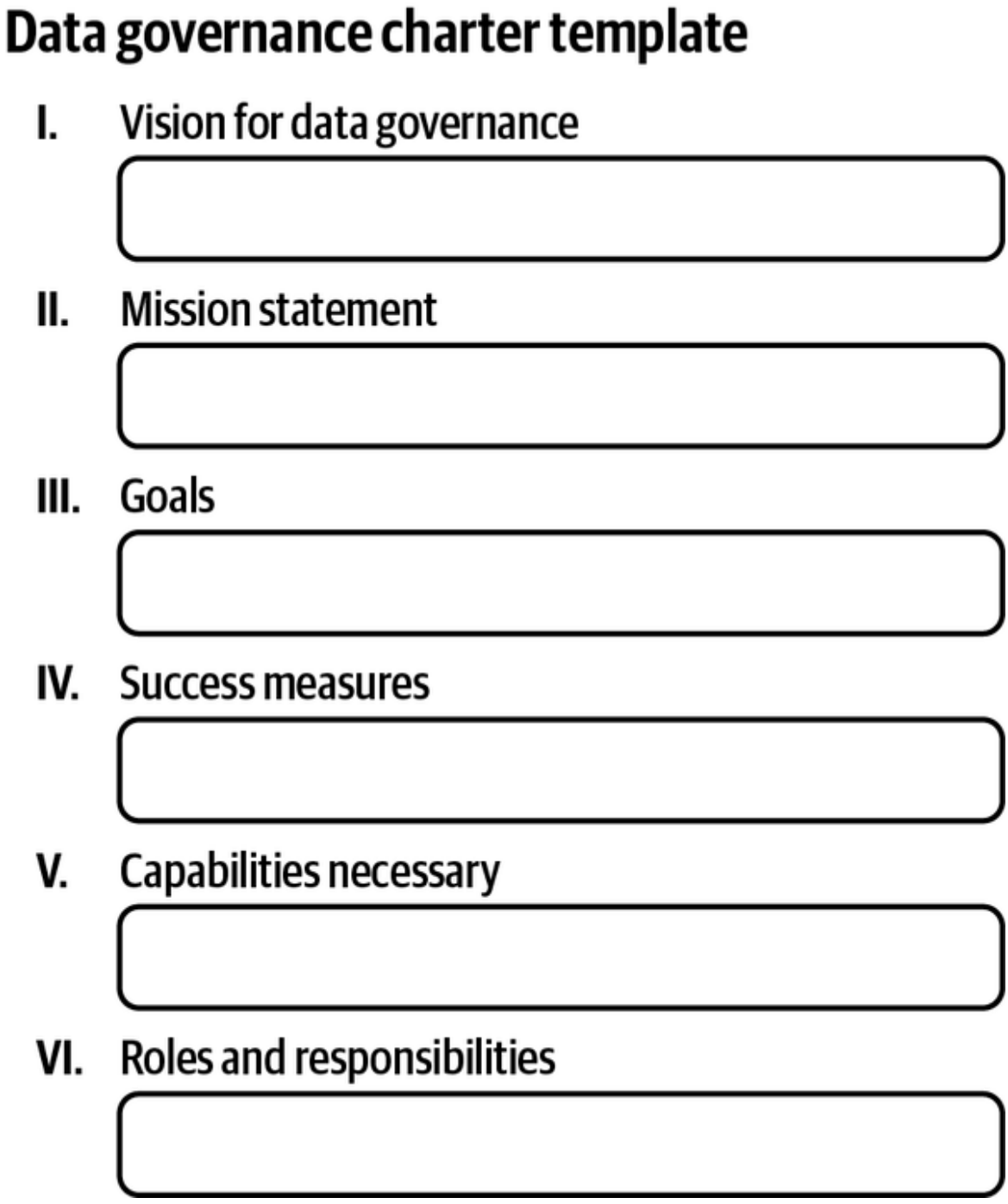


Figure 3-5. Data governance charter template

Importance of a Data Governance Policy

When you have a business idea and are going to friends to socialize the idea and possibly get them to buy into it, you will quickly run into someone who asks for a business plan. “Do you have a business plan you can share so I

can read more about this idea and what your plans are?" A data governance policy allows you to have all the important elements of operationalizing governance documented according to your organization's needs and objectives. It also allows consistency within the organization over a long period of time. It is the document that everyone will refer to when questions and issues arise. It should be reviewed regularly and updated when things in the organization change. You can consider it your business plan—or to another extreme, it can also be your governance bible.

When a data governance policy is well drafted, it will ensure:

- Consistent, efficient, and effective management of the data assets throughout the organization and data life cycle and over time.
- The appropriate level of protection of the organization's data assets based on their value and risk as determined by the data governance committee.
- The appropriate protection and security levels for different categories of data as established by the governance committee.

Developing a Data Governance Policy

A data governance policy is usually authored by the data governance committee or appointed data governance council. This committee will establish comprehensive policies for the data program that outline how data will be collected, stored, used, and protected. The committee will identify risks and regulatory requirements and look into how they will impact or disrupt the business.

Once all the risks and assessments have been identified, the data governance committee will then draft policy guidelines and procedures that will ensure the organization has the data program that was envisioned. When a policy is well written, it helps capture the strategic vision of the data program. The vision for the governance program could be to drive digital transformation for the organization, or possibly to get insights to drive new revenue or even to use data to provide new products or services.

Whichever is the case for your organization, the policies drafted should all coalesce toward the articulated vision and mission as outlined in the data governance charter template.

Part of the process of developing a data governance policy is establishing the expectations, wants, and needs of key stakeholders through interviews, meetings, and informal conversations. This will help you get valuable input, but it's also an opportunity to secure additional buy-in for the program.

Data Governance Policy Structure

A well-crafted policy should be unique to your organization's vision, mission, and goals. Don't get hung up on every single piece of information on this template, however; use it more like a guide to help you think things through. With that in mind, your governance policy should address:

Vision and mission for the program

If you used a data governance charter template as outlined in Figure 3-5. to get buy-in from other stakeholders, that means you already have this information readily available. As mentioned before, the vision for the governance program could be to drive digital transformation for the organization, or to get insights to drive new revenue, or even to use data to provide new products or services.

Policy purpose

Capture goals for your organization's data governance program, as well as metrics for determining success. The mission and vision of the program should drive the goals and success metrics.

Policy scope

Document the data assets covered by this governance policy. In addition, inventory the data sources and determine data classifications based on whether data is sensitive, confidential, or publicly available, along with the levels of security and protection required at the different levels.

Definitions and terms

The data governance policy is usually viewed by stakeholders across the organization who might not be familiar with certain terms. Use this section to document terms and definitions to ensure everyone is on the same page.

Policy principles

Define rules and standards for the governance program you're looking to set up along with the procedures and programs to enforce them. The rules could cover data access (who has access to what data), data usage (how the data will be used and details around what's acceptable), data integration (what transformations the data will undergo), and data integrity (expectations around data quality). Develop best practices to protect data and to ensure regulations and compliance are effectively documented.

Program structure

Define roles and responsibilities (R&Rs), which are positions within the organization that will oversee elements of the governance program. A RACI chart could help you map out who is responsible, who is accountable, who needs to be consulted, and who should be kept informed about changes. Information on governance R&Rs can be found in Chapter 3 of the book.

Policy review

Determine when the policy will be reviewed and updated and how adherence to the policy will be monitored, measured, and remedied.

Further assistance

Document the right people to address questions from the team and other stakeholders.

It's not enough to document a data governance policy as outlined in Figure 3-6 communicating it to all stakeholders is equally important. This could happen through a combination of group meetings and training, one-on-one conversations, recorded training videos, and written communication.

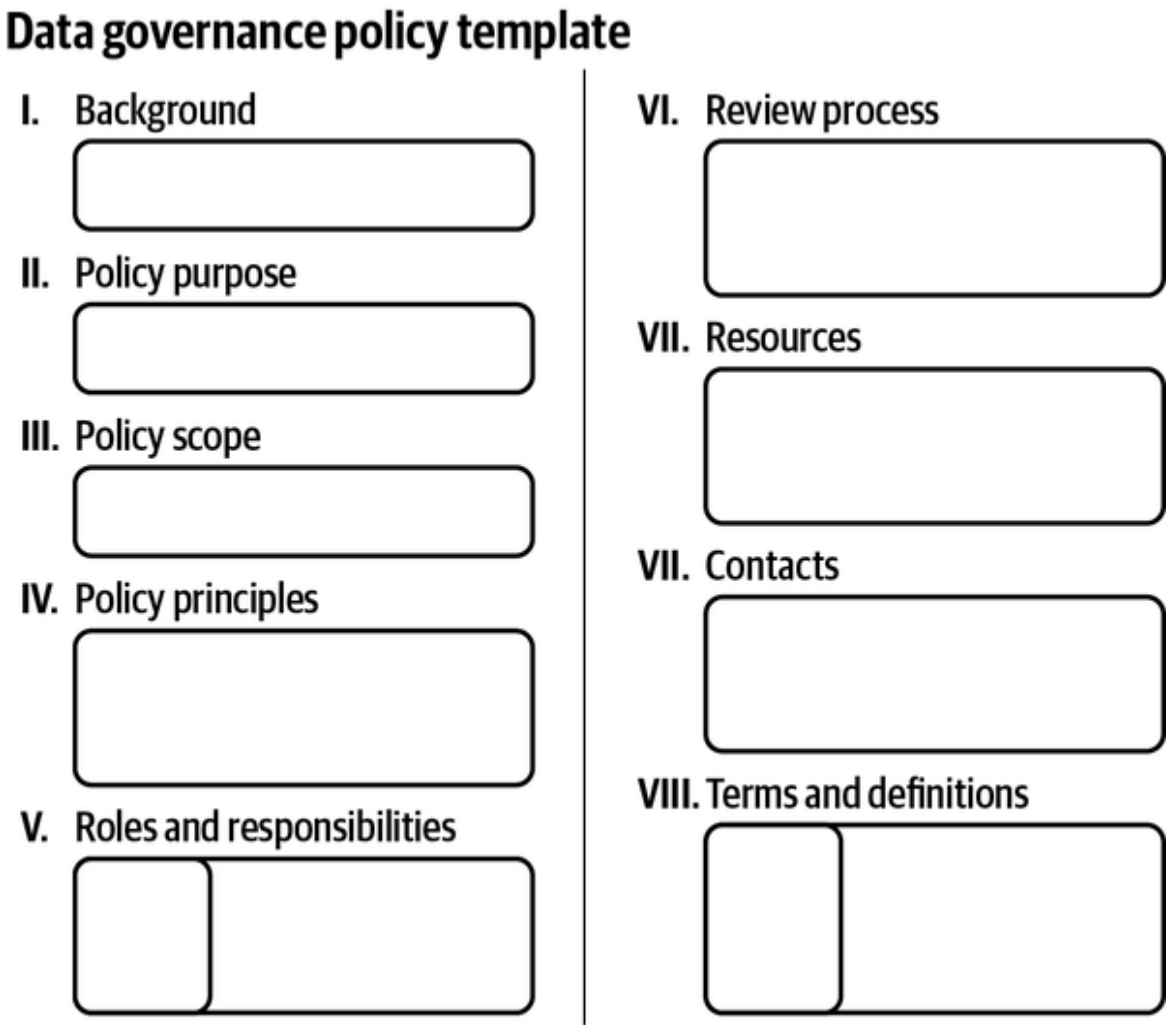


Figure 3-6. Example data governance policy template

In addition, review performance regularly with your data governance team to ensure that you're still on the right track. This also means regularly reviewing your data governance policy to make sure it still reflects the current needs of the organization and program.

Roles and Responsibilities

When operationalizing governance over a data life cycle, you will interact with many stakeholders within the organization, and you will need to bring them together to work on this common goal. While it might be tempting to definitively say which roles do what at which part of the data life cycle, as outlined in Chapter 3, many data governance frameworks revolve around a complex interplay of roles and responsibilities. The reality is that most

companies rarely are able to exactly or fully staff governance roles due to lack of employee skill set or, more commonly, a simple lack of headcount. For this reason, employees working in the information and data space of their company often wear different user “hats.”

We will not go into detail about roles and responsibilities in this chapter, because they’re well outlined in Chapter 3. You still need to define what these look like within your organization and how they will interplay with each other to make governance a reality for you. This will typically be outlined in a RACI matrix describing who is “responsible, accountable, to be consulted, and to be informed” within a certain enforcement, process, policy, or standard.

Step-by-Step Guidance

By this section of the book, you should know that data governance goes beyond the selection and implementation of products and tools. The success of a data governance program depends on the combination of people, processes, and tools all working together to make governance a reality. This section will feel very familiar, because it gathers all the elements discussed in the previous section on data governance policy and puts them in a step-by-step process to show you how to get started. It further double clicks into the concepts as well.

Build the business case

As previously mentioned, data governance takes time and requires investment in people and technology.. If done correctly, it can be automated as part of the application design done at the source with a focus on business value. That said, data governance initiatives will often vary in scope and objectives. Depending on where the initiative is originating from, you need to be able to build a business case that will identify critical business drivers and justify the effort and investment of data governance. It should identify the pain points, outline perceived data risks, and indicate how governance helps the organization mitigate those risks and enable better business outcomes. It’s OK to start small, strive for quick wins, and build up

ambitions over time. Set clear, measurable, and specific goals. You cannot control what you cannot measure; therefore you need to outline success metrics. The data governance charter template in Figure 3-6 is perfect for helping you get started.

Document guiding principles

Develop and document core principles associated with governance and, of course, associated with the project you're looking to get off the ground. A core principle of your governance strategy could be to make consistent and confident business decisions based on trustworthy data aligned with all the various purposes for the use of the data assets. Another core principle could be to meet regulatory requirements and avoid fines or even to optimize staff effectiveness by providing data assets that meet the desired data quality thresholds. Define principles that are core to your business and project. If you're still new to this area, there are a lot of resources available. If you are looking online, there are several vendor-agnostic, not-for-profit associations, such as the Data Governance Institute (DGI), the Data Management Association (DAMA), the Data Governance Professionals Organization (DGPO), and the Enterprise Data Management Council, all of which provide great resources for business, IT, and data professionals dedicated to advancing the discipline of data governance. In addition, identify whether there are any local data governance meetup groups or conferences that you can possibly attend, such as the Data Governance and Information Quality Conference, DAMA International Events, or a Financial Information Summit.

Get management buy-in

It should be no surprise that without management buy-in, your governance initiative can easily be dead from the get-go. Management controls the big decisions and funding that you need. Outlining important KPIs, and how your plan helps to move them, will get management to be all ears. Engage data governance champions and get buy-in from the key senior stakeholders. Present your business case and guiding principles to C-level management for approval. You need allies on your side to help make the

case. And once the project has gotten off the ground, communicate frequently.

Develop an operating model

Once you have management approval, it's time to get to work. How do you integrate this governance plan into the way of doing business in your enterprise? We introduced you to the data governance policy, which can come in very handy during this process. During this stage, define the data governance roles and responsibilities, and then describe the processes and procedures for the data governance council and data stewardship teams who will define processes for defining and implementing policies as well as for reviewing and remediating identified data issues. Leverage the content from the data management policy plan to help you define your operating model. Data governance is a team sport, with deliverables from all parts of the business.

Develop a framework for accountability

As with any project you're looking to bring to market, establishing a framework for assigning custodianship and responsibility for critical data domains is paramount. Define ownership. Make sure there is visibility to the "data owners" across the data landscape. Provide a methodology to ensure that everyone is accountable for contributing to data usability. Refer back to your data management policy, as it probably started to capture some of these dependencies.

Develop taxonomies and ontologies

This is where a lot of the education you've collected thus far comes in handy. Working closely with governance associations, leaning in on your peers, and simply learning about things online will help you with this step. There may be a number of governance directives associated with data classification, organization, and, in the case of sensitive information, data protection. To enable your data consumers to comply with those directives, there must be a clear definition of the categories (for organizational

structure) and classifications (for assessing data sensitivity). These should be captured in your data governance policy.

Assemble the right technology stack

Once you've assigned data governance roles to your staff and defined and approved your processes and procedures, you should then assemble a suite of tools that facilitates implementation and ongoing validation of compliance with data policies and accurate compliance reporting. Map infrastructure, architecture, and tools. Your data governance framework must be a sensible part of your enterprise architecture, the IT landscape, and the tools needed. We talked about technology in previous sections, so we won't go into detail about it here. Finding tools and technology that work for you and satisfy the organizational objectives you laid out is what's important.

Establish education and training

As highlighted earlier, for data governance to work, it needs buy-in across the organization. You need to ensure that your organization is keeping up and is still buying into the project you presented. It's therefore important to raise awareness of the value of data governance by developing educational materials highlighting data governance practices, procedures, and the use of supporting technology. Plan for regular training sessions to reinforce good data governance practices. Wherever possible, use business terms, and translate the academic parts of the data governance discipline into meaningful content in the business context.

Considerations for Governance Across a Data Life Cycle

Data governance has been around since there was data to govern, but it was mostly viewed as an IT function. Implementing data governance across the data life cycle is not trivial. Here are some considerations you will need to think about as you implement governance in your organization.

Deployment time

Crafting and setting up governance processes across the data life cycle takes a lot of time, effort, and resources. In this chapter, we have introduced a lot of concepts, ideas, and ways to think about operationalizing governance across the data life cycle, and you can see it gets overwhelming very quickly. There's not a one-size-fits-all solution; you need to identify what is unique about your business and then forge a plan that works for you. For most organizations, automation is critical for a governance program to scale. Leveraging AI can greatly accelerate governance implementations for everything from autoclassification to anomaly detection but they must be well monitored. That means that as you look for solutions in the market, you will need to find out how much automation and integration is built into it, how well it works for your environment and situation, and whether that is the most difficult part of the workflow that could use automation. In a multi-cloud world, this can become even more complex and further increase implementation time.

Complexity and cost

Complexity comes in many forms. In Chapter 1, we talked about how large and varied the data landscape is and the increasing rate data is produced. Another complexity is a lack of defined industry standards for things like metadata. We touched on this in Chapter 2. In most cases, metadata does not obey the same policies and controls as the underlying data itself, and a lack of standardized metadata specifications means that different products and processes will have different ways of presenting this information. Still another complexity is the sheer amount of tools, processes, and infrastructure needed to make governance a reality. In order to deliver comprehensive governance, organizations must either integrate best-of-breed solutions, which are often complex and very expensive (with high license and maintenance costs), or buy turnkey, integrated solutions, which are expensive and fewer in the market or build their own. With this in mind, cloud service providers (CSPs) and the large data platforms are providing all these governance capabilities built in, thus creating a one-stop shop and simplifying the process for customers. As an organization, research and compare the different offerings and see which work for you. Selecting

partners and tools that can operate in multiple cloud environments or are committed to supporting multi-cloud environments will help reduce complexity and the risk of costly lock-in.

Changing regulation environment

In previous chapters, we've clearly outlined the implications of a constantly changing regulatory environment with the introduction of GDPR, CCPA and EU AI Act. We will not go into the same detail here; however, regulations define a lot of what must be done and implemented to ensure governance. They will outline how certain types of data need to be handled and which types of controls need to be in place, and they sometimes will even go as far as outlining what the repercussions are when these things are not complied with. Complying with regulations is absolutely something your organization needs to think about as you implement data governance over the data life cycle.

> **NOTE**
>
> In our discussions with many different companies, we've heard of two very different philosophies when it comes to considering changes to the regulatory environment. One strategy is to assume that, in the future, the most restrictive regulations that are present now will cascade and be required everywhere (like CCPA being required across the entire US and not just in California), and that ensuring compliance now, even though not required, is a top priority. Conversely, we've also heard the strategy of complying only with what's required right now and dealing with regulations only if they become required. We strongly suggest you take the former approach, because a proper and well-thought-out governance program not only ensures compliance with ever-changing regulations; it also enables many of the other benefits we've outlined thus far, such as better findability, better security, and more accurate analytics from higher-quality data.

Location of data

In order to fully implement governance over a data life cycle, understanding which data is on-premises versus in the cloud is very important. Furthermore, understanding how data will interact with other data along the life cycle does create complexity. In the current paradigm, most organizational data lives both on-premises and in the cloud, and having

systems and tools that allow for hybrid and even multicloud scenarios is paramount. In Chapter 1, we talked about why governance is easier in the public cloud—it's primarily because the public cloud has several features that make data governance easier to implement, monitor, and update. In many cases, these features are unavailable or cost-prohibitive in on-premises systems. Data should be protected no matter where it is located, so a viable data life cycle management plan will incorporate governance for all data at all times.

Organizational culture

As you know, culture is one of those intangible things in an organization that plays an important role in how the organization functions. In Chapter 3, we touched on how an organization can create a culture of privacy and security, which allows employees to understand how data should be managed and treated so that they are good stewards of proper data handling and usage.

In this section, we're referring to organizational culture, which often dictates what people do and how they behave. Your organization might be free, allowing folks to easily raise questions and concerns, and in such an environment, when something goes wrong, people are more likely to speak up. In organizations in which people are reprimanded for every little thing, they will be more afraid to speak up and report when things are not working or even when things go wrong. In these environments, governance is difficult to implement, because without transparency and proper reporting, mistakes are usually not discovered until much later. In the NASA example we provided earlier in this chapter, there were a couple of people within the organization who noticed the discrepancy in the data and even reported it. Their reports were ignored by management, and we all know what happened. Things did not end well for NASA. Remember, instituting governance in an organization is often met with resistance, especially if the organization is accustomed to decentralized operations. Creating an environment in which functions are centralized across the data life cycle simply means that these areas have to adhere to processes that they might

not have been used to in the past but that are for the larger good of the organization.

Summary

Data life cycle management is paramount to implementing governance and ensures that useful data is clean, accurate, and readily available to users and agents. In addition, it ensures that your organization remains compliant at all times.

In this chapter, we introduced you to data life cycle management, and how to apply governance over the data life cycle. We then looked into operationalizing governance and how the role of a data governance policy is to ensure that an organization's data and information assets are managed consistently and used properly. Finally, we provided step-by-step guidance for implementing governance and finished with the considerations for governance across the data life cycle, including deployment time, complexity and cost, and organizational culture.

Chapter 4. Governance of Data in Flight

> **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 6th chapter of the final book. Please note that the GitHub repo will be made active later on.
>
> If you'd like to be actively involved in reviewing and commenting on this draft, please reach out to the editor at *ccollins@oreilly.com*.

Data, especially data used for insights via data analytics, is a "living" medium. As data gets collected from multiple sources, it is reshaped, transformed, and molded into various patterns for different use cases: from a standardized "transactions table" to allow for forecasting the next season's business demand, to a dashboard presenting the past yield of a new crop, and more.

Data governance should be consistent across these transformations and allow more efficiency and frictionless security. Data governance should not introduce labor by forcing users to register and annotate new data containers as they work to reshape and collect data for their needs.

This chapter will discuss possible techniques and tools to enable seamless data governance through analysis of data "in flight."

Data Transformations

There are different ways to transform data, all of which impact governance, and we should be aware of these before we dig in deeper. It is common to refer to these processes as extract-transform-load (ETL). This is a generic phrase used to indicate the various stages of moving data between systems.

Extracting data means retrieving it from the source system in which it is stored, e.g., a legacy DB, a file, or the results of a web crawler operation. Data extraction is a separate step, as the act of extracting data is a time-consuming retrieval process. It is advantageous to consider the extraction phase as the first step in a pipeline, allowing subsequent steps to operate in batches in parallel to the continued extraction. As data is extracted from the sources, it's useful to perform *data validation*, making sure the values retrieved are "as expected" (that the completeness of records and their accuracy match the expected values (see Chapter 5). If you perform data validation while still operating within the context of the source system, you will be unencumbered by the different computed results performed in later stages, which may be unknown (at this stage), and as you progress, you may lose the context of the source data. In an earlier chapter, we discussed data preparation, which is an example of a data validation process. The data being extracted and validated often lands in a *staging area* not normally accessible to the business users, which is where the data owners and data stewards perform the aforementioned validation checks.

Transforming data usually involves normalization of the data: eliminating outliers, joining from multiple sources into a single record (row), aggregating where relevant, or even splitting a single compound column into multiple columns. Be wary of the fact that any normalization done early, as well as any kind of general purpose cleaning, is also removing information—information whose value may not have been anticipated for a case unknown at the cleaning level. The implications are that you should have the business context in mind when extracting data and that you may need to revisit the source data in case the extraction process removed information needed for a new, unforeseen use case. At this stage, if you are extracting data from a new source, it may be worthwhile to create a scorecard for the source, describing some of the information contexts and

those that are potentially not carried over during transformation. See Chapter 5 for more about scorecards.

Finally, the load process situates the data into its final destination, usually a data-analytics-capable warehouse such as Google's BigQuery, Snowflake, or Amazon's Redshift.

It is important to note that as data warehousing solutions grew to be more powerful, the transformation process sometimes moved into the data warehousing solution, renaming ETL as ELT.

As data undergoes transformations, it is crucial not only to keep context as expressed by the origin but also to maintain consistency and completeness. Keeping origin context, consistency, and completeness will allow a measure of trustworthiness in the data, which is critical for data governance.

GitOps for Data Resilience: Moving Beyond Static Staging

In the first edition, we discussed the necessity of staging areas—isolated environments where data is transformed and validated before reaching production. We can't just govern where data lands anymore; we have to govern how it gets there. Historically, data transformations happened in static staging areas—black boxes that business teams couldn't see, isolated from the core application, and completely lacking real version control. That doesn't work in an AI-driven environment. Today, a single silent corruption or an unexpected schema drift in a streaming pipeline will instantly poison downstream vector databases and live agent contexts. To prevent that, we need to bring active GitOps principles right into the streaming pipeline.

To govern data in flight effectively in the era of AI, we recommend adopting a GitOps approach to data resilience. This approach treats data pipelines, transformation logic, and schemas as version-controlled code, allowing organizations to manage the data lifecycle with the same rigor used in software engineering.

The Evolution of the Data Pipeline

Modern resilience comes down to three fundamental shifts in how we handle data while it is moving:

From Static to Versioned Branches

> We need to stop moving data into fixed, physical staging databases. High-maturity organizations are moving to "branching" architectures instead. This lets engineers spin up a virtual branch of a live production stream, test a new transformation or schema change against real-time data, and verify the output without ever touching the primary production flow.

Immutable Transformation Logic

> We have to eliminate the old "black box" transformations where data quality degrades silently over time. By storing transformation logic directly in a version-control system, every single change to how data is cleaned or aggregated is logged, audited, and tied straight back to a specific governance policy.

The Rollback Mandate

> Governance isn't just about preventing errors anymore—it's about how fast you can recover. When an unexpected schema change or a quality anomaly pops up over the wire, the system has to trigger an automated rollback to the last known-good state. This is the only way to shield downstream AI agents and analytics dashboards from poisoned data and keep the enterprise asset trustworthy.

NOTE

Adopting GitOps for data doesn't mean governance gets handed over to a technical silo. Actually, it does the opposite. It gives business leaders a completely transparent change log of how data has evolved, showing them exactly why a transformation happened and when the pipeline was last verified for compliance.

The Concept of Data Branching

Traditional governance focused on fixed validation points right at the source. A GitOps approach changes this by letting architects spin up a “branch” of a live streaming pipeline. This gives teams two major advantages:

Live Testing

: You can test new schema updates, compliance filters, or normalization rules on a live, completely isolated branch without risking production stability.

Pre-Merge Validation

: It forces an automated check to prove the new data path meets corporate standards for trust and completeness before it ever touches the main data flow.

The Necessity of Rollbacks

In the original text, we discussed the importance of data lineage and audit logs to understand where data came from. Lineage and audit logs are great for figuring out what went wrong after the fact, but modern architectures require proactive containment. If a data quality issue or an unauthorized change happens over the wire, you need to hit undo instantly.

Treating Data as Code

: Modern data versioning tools let organizations treat their operational data state exactly like software code.

Instant Recovery

: If an automated transformation fails, the system immediately rolls the pipeline back to the last known-good state. This stops degraded or biased data from hitting downstream AI models, protecting the integrity of the enterprise asset before the damage spreads.

This kind of automated resilience sounds great on paper, but you can't roll back a pipeline or branch a live data stream if you are flying blind. To make GitOps work in practice, the system has to know exactly where data came from, what happened to it the moment it hit the wire, and where it is headed next. This brings us squarely to the operational backbone of data in flight: real-time lineage and metadata tracking.

Tracking where data flows and capturing its lineage is only half the battle. In a modern architecture, we aren't just logging this metadata for historical audits; we need to use it to actively protect user trust. This becomes incredibly clear when you look at the operational nightmare of managing user privacy. It's one thing to have a static policy on a corporate server, but it's another thing entirely to enforce that choice when a user changes their mind mid-stream.

Dynamic Consent Lineage: Handling the Opt-In/Opt-Out Lifecycle in Flight

As we noted in the earlier chapters, managing user trust is a non-negotiable part of modern data architecture. But there is a massive difference between setting a corporate privacy policy and actually enforcing it when data is blasting through a streaming pipeline at scale.

Historically, data governance treated consumer consent like a static checkpoint—a simple check-box in a customer profile database or a fixed marketing exclusion list updated on a batch schedule. That old way completely breaks down in a world driven by real-time retrieval and continuous AI training loops. If a user changes their mind and switches from "Opt-In" to "Opt-Out," you cannot afford to wait for a weekly batch job to scrub their data. By then, that data has already flown down the wire, slipped into a vector space, or been absorbed into an active model context window.

To maintain real trust, consent has to become a fluid, active attribute that travels *with* the data over the wire.

Consent as Immutable Streaming Metadata

We have to stop looking at consent as a static row in a database table and start treating it as a permanent, immutable tag bound directly to the data packet while it is in flight.

The Travel Companion

> The moment an interaction or event is captured, its current consent status must be baked right into the payload's metadata wrapper.

Inheritance Across the Wire

> As that data moves through various real-time transformations, aggregations, or streaming topics, the pipeline must force downstream payloads to automatically inherit those exact security and privacy classifications. If the source data packet is flagged as restrictive, any transformed version of it downstream inherits that exact restriction.

Real-Time Purge and Exclusion Triggers

The real test of a data-in-flight architecture isn't how it handles data ingestion; it's how fast it can execute an undo. When a user hits "Opt-Out," that preference change should be treated as a high-priority, real-time event that streams through the exact same architecture.

Downstream Interception

> The moment the opt-out event hits the wire, the pipeline needs to act as an automated gatekeeper, instantly triggering downstream purges or exclusions.

Shielding the AI Space

> This real-time interception ensures that the revoked data is immediately dropped or bypassed *before* it can feed into downstream vector indices, RAG pipelines, or active training checkpoints.

By building consent right into the streaming metadata, we move privacy out of the legal department's spreadsheets and embed it directly into the engineering foundation of the pipeline, ensuring compliance is enforced at the exact same speed the business moves.

Lineage

Following the discussion of data transformations, it is important to note the role played by data lineage. Lineage, or *provenance*, is the recording of the "path" that data takes as it travels through extract-transform-load, and other movement of data, as new datasets and tables are created, discarded, restored, and generally used throughout the data life cycle. Lineage can be a visual representation of the data origins (creation, transformation, import) and should help answer the questions "Why does this dataset exist?" and "Where did the data come from?"

Why Lineage Is Useful

As data moves around your data lake, it intermingles and interacts with other data from other sources in order to produce *insight*. However, metadata—the information about the data sources and their classification—is at risk of getting lost as data travels. For example, you can potentially ask, for a given data source, "What is the quality of the data coming from that source?" It could be a highly reliable automatic process, or it could be a human-curated/validated dataset. There are other sources that you potentially trust less. As data from different sources intermingle, this information can sometimes be lost. Sometimes it is even desirable to forgo mixing certain data sources in order to preserve authenticity.

In addition to data quality, another common signal potentially available for source data is *sensitivity*. Census information and a recently acquired client phone list are of a certain level of sensitivity, while data scraped off a publicly available web page may be of a different sensitivity.

Thus, data sensitivity and quality, and other information potentially available on the origin, should filter down to the final data products.

The metadata information of the data sources (e.g., sensitivity, quality, whether or not the data contains PII, etc.) can support decisions about whether or not to allow certain data products to mix, whether or not to allow access to that data and to whom, and so on. When mixing data products, you will need to keep track of where the data came from. The business purpose for the data is of particular importance, as the created data products should be useful in achieving that purpose. If the business purpose requires, for example, a certain level of accuracy for time units, make sure the lineage of your data does not coarsen time values as the data is processed. Lineage is therefore crucial for an effective data governance strategy.

How to Collect Lineage

Ideally, your data warehouse or data catalog will have a facility that starts from the data origin and ends with whatever data products, dashboards, or models are used, and that can collect lineage for every transaction along the way. This is far from common, and there will likely be blindspots. You will have to either infer the information that you need on by-products or otherwise manually curate information to close the gap. Once you have this information, depending on how trustworthy it is, you can use it for governance purposes.

As your data grows (and it is common for successful businesses to accumulate data at an exponential rate) it is important to allow more and more automation into the process of lineage collection and to rely less and less on human curation. Automation is important because, as we will see in this chapter, lineage can make a huge difference in data governance, and placing human bottlenecks in the path to governance blocks an organization’s ability to make data accessible. Also, a broken chain of lineage (e.g., due to human error) can have a larger impact on derivative data products, as trust becomes harder to come by.

Another way to collect/create lineage information is to connect to the API log for your data warehouse. The API log is expected to contain all SQL jobs and also all programmatic pipelines (R, Python, etc.). If there is a good job audit log, you can use it to create a lineage graph. This allows backtracking table creation statements to the table's predecessors, for example. This is not as effective as just-in-time lineage recording, as it requires backtracking the logs and batch processing. But assuming (not always true!) that you are focusing on lineage within the data warehouse, this method can be extremely useful.

Types of Lineage

The level of granularity of the lineage is important when discussing possible applications for that lineage. Normally, you would want lineage at least in the table/file level—i.e., this table is a product of this process and this other table.

In Figure 4-1, we see a very simple lineage graph of two tables joined by a SQL statement to create a third table.

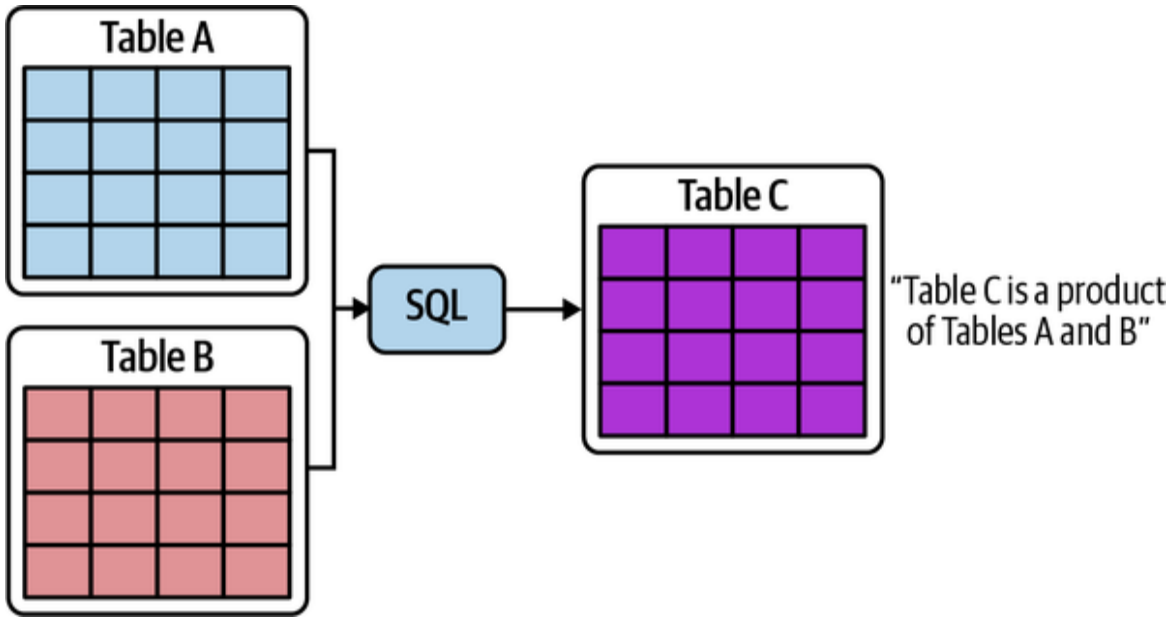


Figure 4-1. Table-level lineage

A column/field-level granularity is more useful: "This table consists of the following columns from this other table and another column from that

table.” When talking about column-level granularity, you can start to talk about specific types of data being tracked. In structured (tabular) data, a column is normally only a single data type. An example business case would be tracking PII: if you mark at the sources which columns are PII, you can continue tracking this PII as data is moved and new tables are created, and you can confidently answer which tables have PII. In Figure 4-2, two columns from Table A are joined with two columns from Table B to create Table C. If those source columns were “known PII” (as an example), you can use lineage to determine that Table C now contains PII as well.

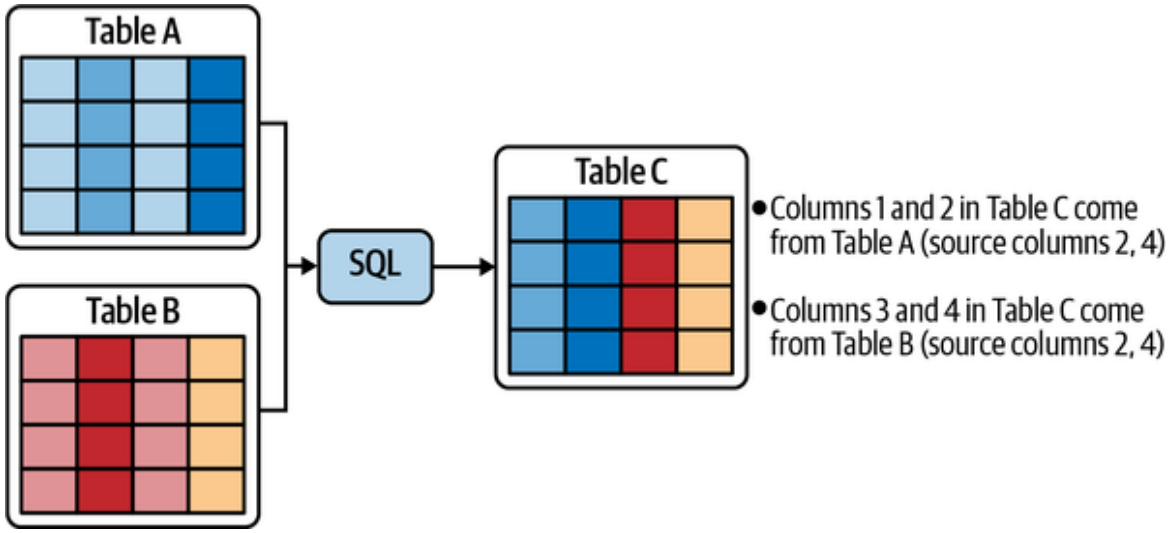


Figure 4-2. Column-level lineage

Row-level lineage allows the expression of information about transactions. Dataset-level lineage allows the expression of coarse information about data sources.

MORE GRANULAR ACCESS CONTROLS

One of the most common use cases we have heard during our research and interviews is that, while project/file-level access controls work, being able to more granularly control access is highly desired. For example, column-level access control ("Provide access to columns 1–3 and 5–8 of this table but not column 4") allows you to lock down, if you will, a particular column of data and still allow access to the rest of the table.

This method works especially well for tables that contain much usable, relevant analytics data but may also contain some sensitive data. A prime example of this would be a telecommunications company and its retail store transactions. Each retail store's transaction logs not only would contain information about each item purchased, along with date, price, etc., but also would likely contain the purchaser's name (in the event the purchaser is a customer of the telecom company and put the item onto their account). An example system leveraging labels-based (or attribute-based) granular access controls is depicted in Figure 4-3.

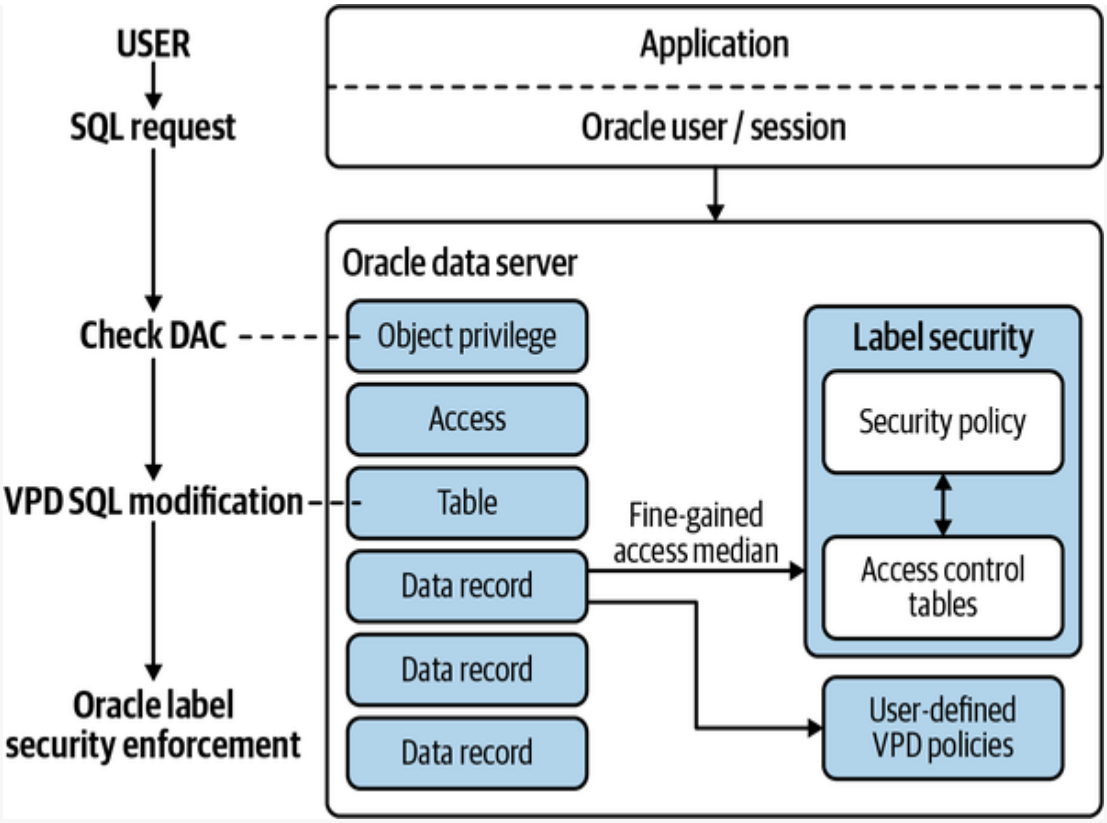


Figure 4-3. Label-based security in Oracle

An analyst certainly wouldn't need the customer's name or account information, but they would need the other relevant information of item purchased, location, time, price, and so on. Instead of having to either grant full access to this table or rewrite the table with the sensitive information removed, the sensitive column itself can have an access control so that only certain individuals are allowed access, and all others simply see the table with that column removed or redacted/hashed/encrypted in some way.

As we talked about in Chapter 3, most companies do not have enough headcount to support the constant monitoring, tagging, and rewriting of tables to remove or restrict sensitive information. Granular access controls can enable you to get the same result (sensitive information is guarded) while also allowing greater data democratization.

The Fourth Dimension

As data gets created, coarsened, and discarded, it is important to realize that lineage is a temporal state—while a certain table is currently a product of other tables, it is possible that in a previous iteration the very same table was generated out of other data.

Looking at the "state of now" is useful for certain applications, which we will discuss shortly, but it is important to remember that the past versions of certain data objects are relevant to gaining a true understanding of how data is being used and accessed throughout an enterprise. This requires versioning information that needs to be accessible on the object.

How to Govern Data in Flight

Working off the assumption that lineage information is preserved and is reliable to a certain extent, there are key governance applications that rely on lineage.

A common need that gets answered by lineage is debugging or understanding sudden changes in data. Why has a certain dashboard stopped displaying correctly? What data has caused a shift in the accuracy of a certain machine learning algorithm? And so on. Finding from an end product which information feeds into that end product and looking for changes (e.g., a sudden drop in quality, missing fields, unavailability of certain data) could significantly speed up tasks related to end products. Understanding the path and transformations of the data can also help troubleshoot data transformation errors.

Another need that commonly requires at least field-level lineage (and said lineage needs to be reliable) is the ability to infer data-class-level policies. For a certain column that contains PII, I want to mark all future descendants of this column as PII and to effect the same access/retention/masking policies of the source column to the derivatives. You will likely need some algorithmic intelligence that allows you to effect an identical policy if the column is copied precisely and to effect a different (or no) policy if the column contents are nullified as a result of the creation of a derivative.

In Figure 4-4, we see a slightly more involved lineage graph. While the operations performed to create derivatives are marked as "SQL," note that this is not always the case; sometimes there are other ways to transform the data (for example, different scripting languages). As data moves from the external data source into Table D, and as Table D is merged with Table C (the by-product of Tables A and B) and finally presented in a dashboard, you can see how keeping lineage, especially column-level lineage, is important.

For example, a question asked by the business user would be "Who can I show this dashboard to?" Let's say that just one of the sources (Table A, Table B, the external data source) contains PII in one of the columns, and that the desired policy on PII is "available only to full-time employees"; if PII made it into the dashboard, that should effect an access policy on the dashboard itself.

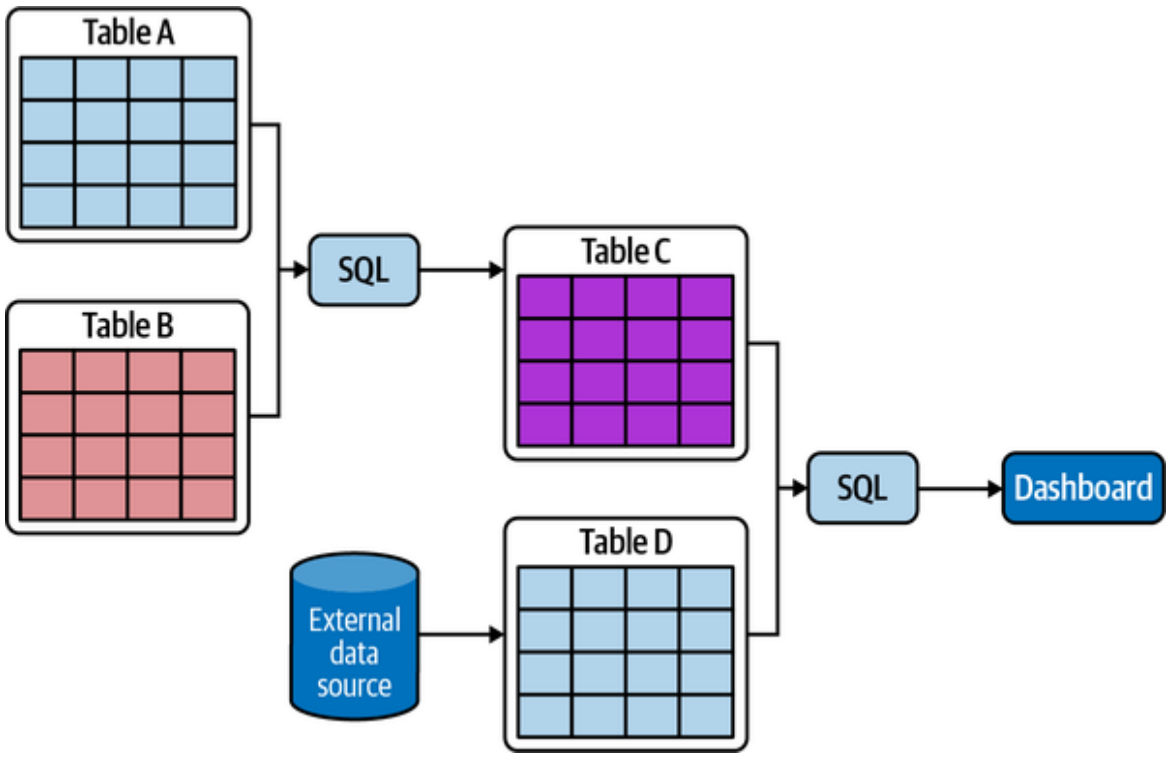


Figure 4-4. Lineage workflow—if Table B contains sensitive data, that data can potentially be found in the dashboard as well

These use cases are rather specific, but broader use cases have been expressed by CIOs and CISOs for a long time:

"Show me all the sensitive data within our data warehouse, and what systems contain sensitive data."

> This is a simple-to-express ask, and with the combination of lineage and the ability to classify data sources, it is much simpler to answer than understanding the multitude of data objects and systems that may be part of an organization's data lake.

"I want to identify certain systems as trusted, and make sure data does not exist in other, less trusted systems without manual oversight."

> This need can be addressed by enforcing egress controls on the trusted systems, a task which is potentially simpler with a good lineage solution. In this way, you can, for example, eliminate data being ingested from unapproved systems, because those sources will show up in the lineage information.

"I need to report and audit all systems that process PII."

> This is a common need in a post-GDPR world, and if you mark all sources of PII, you can leverage the lineage graph to identify where PII gets processed, allowing a new level of control.

Policy Management, Simulation, Monitoring, Change Management

We already provided one example for policy management: *inheritance*. In essence, data governance policies should be derived from the meaning of the data. If in a certain organization we want to govern PII, and PII is defined as all of personal phone numbers, email addresses, and street addresses, these individual infotypes can be automatically detected and associated with the data class. However, scanning a table and determining which columns contain these infotypes is expensive computationally. To correctly identify infotypes, the system will need to sample the relevant columns (without foreknowledge about which columns are sensitive, the

entire table will need to be sampled), process those through a pattern matching and/or a machine learning model that will provide a level of confidence as to the underlying infotype, and tag the columns appropriately.

However, this process is made much more efficient if you can just identify the event of a column creation without changing in values from an already-tagged column. This is where lineage comes in.

Another use for lineage information is when data change management is considered. Let's imagine that you want to delete a certain table; or alternatively, you want to change the access policy on a data class; or maybe you want to set up a data-retention control that will coarsen the data over a period of time (for example, change the data from "GPS Coordinates" to city/state after 30 days). With lineage, you can follow the affected data to its end products and analyze the impact of this change. Let's say you limit access to a certain number of fields in a table. You can now see which dashboards or other systems use the data and assess the impact. A desirable result will be highlighting the change in access to end users accessing end products, so a warning could be issued at the time of changing the policy, alerting the administrator to the fact certain users will lose access and potentially even allowing a level of drill down so those users can be inspected and a more informed decision can be made.

Audit and Compliance

We often need to point at lineage information to be able to prove to an auditor or a regulator that a certain end product (machine learning model, dashboard, etc.) was fed into by specific and preapproved transactional information. This is necessary because regulators will want to be able to explain the reasoning behind certain decision-making algorithms and make sure these rely on data that was captured according to the enterprise's charter—for example, making sure loan approvals rely only on specific credit information that was collected according to regulations.

It is increasingly common for regulated organizations to be able to prove that even machine learning models are not biased, that decisions that are

derived from data are done so without manipulation, and that there is an unbroken "chain of trust" between properly acquired data sources and the end-user tools (e.g., dashboards, expert systems) that are guiding those decisions. For example, in the credit scenario just described a decision on a new line of credit must be traced exclusively to a list of transactions from a trusted source, without other influences guiding the decision.

> **NOTE**
>
> While it's clearly incredibly important to be able to "prove" compliance in the event of an external audit, these same tools can be used for internal auditing purposes as well. In fact, it's best practice (as we've mentioned several times in earlier chapters) to continually assess and reassess your governance program—how it's going, where it may need to be modified, and how it may or may not need to be updated as regulations and/or business needs change.

Summary

We have seen how collecting data lineage can enable data governance policies, inference, and automation. With a lineage graph, organizations can trace data variations and life cycle, promoting control and allowing a complete picture of the various systems involved in data collection and manipulation. For the business user, governing data while it is "in flight" through lineage allows a measure of trustworthiness to be inherited from trusted sources or trusted processors in the data path. This enriched context of lineage allows matching the right datasets (by sensitivity) to the right people or processes.

While lineage is essentially a technical construct, we should always keep the end business goal in mind. This could be "ensuring decisions are made with high-quality data," in which case we need to show lineage to the result from high-quality sources (and be able to discuss the transformations along the way), or a specific business case such as "tracing sensitive data," as we've already discussed.

The technical lineage and the business lineage use cases discussed here are both important, and we should strive to present the many technical details (e.g., intermediate processing tables) to analysts while at the same time serving a simplified “business view” to the business users.

About the Author(s)

Evren Eryurek, PhD, is a seasoned technology executive with over 30 years of experience pioneering AI, cloud, and data solutions. He previously served as Director of Product Management for Google Cloud's data portfolio, Technical Director in the Google Cloud CTO Office, and Software CTO for GE Healthcare. A graduate of the University of Tennessee with a PhD in Nuclear Engineering, Evren holds over 60 US patents and currently serves as an advisor to VCs and AI startups.

Uri Gilad leads competitive and privacy regulatory strategy for Alphabet's Google Ads organization. During his previous tenure in product leadership at Google Cloud, he cocreated its data governance practice, directed BigQuery security, and served as an EDM Council board member. He has an extensive background in cybersecurity executive roles and holds an M.Sc. from Tel Aviv University and a B.Sc. from the Technion.

Dr. Valliappa ("Lak") Lakshmanan is the cofounder and CTO of Obin AI, a startup building deep-domain AI agents for finance. He previously served as Director for Data Analytics and AI Solutions on Google Cloud—where he cofounded the Advanced Solutions Lab—and was a Research Scientist at NOAA. Lak is a veteran O'Reilly author and the creator of multiple popular Coursera courses.

Dmitry Goodman is an enterprise data engineering leader specializing in building scalable data architectures, modernizing data pipelines, and establishing robust governance frameworks for complex data ecosystems.

Jessi Ashdown is a User Experience Researcher for Google Cloud, where she conducts global user studies to directly shape and inform the design of Google's data governance products. Before joining Google, she led the Enterprise UX Research team at T-Mobile. Jessi holds a bachelor's degree in Psychology from the University of Washington and a master's degree in Human-Computer Interaction from Iowa State University.