MONTE CARLO

Building Data Products:

The Ultimate How-To Guide

Design and ship valuable data products your company will actually use.

Table of Contents

I. **Introduction**
- What is a data product?
- What are the benefits?

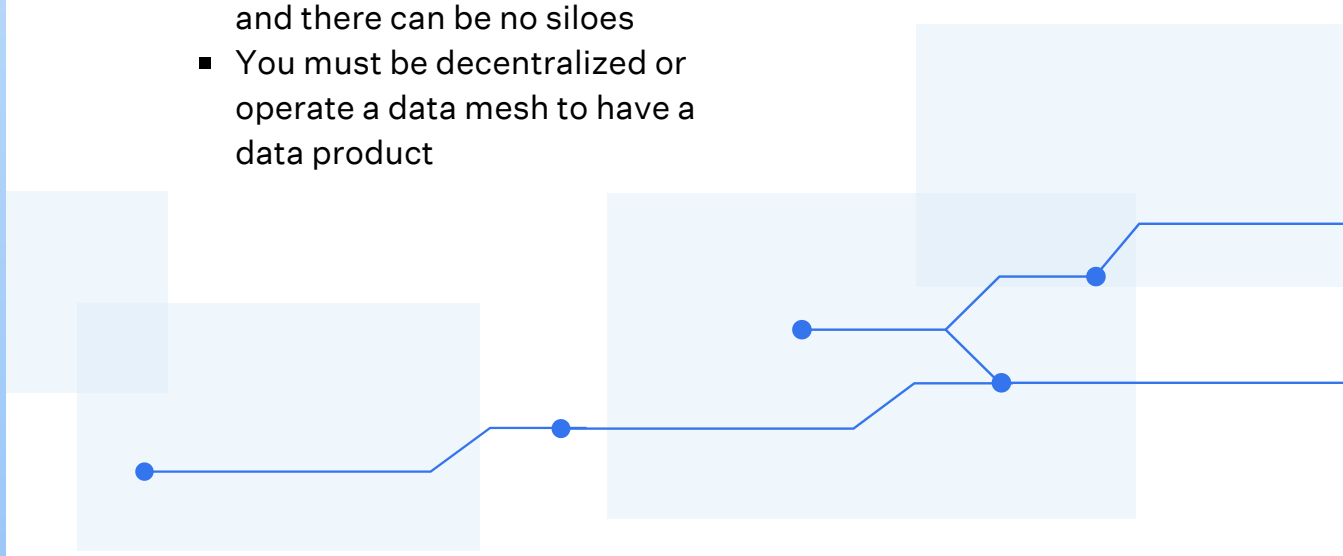
II. **Designing a Data Product**
- Creating SLAs
- Assigning ownership
  - Organizational ownership
  - Project ownership
  - Table/pipeline ownership
- Documentation, discovery, and self-service
- Governance and compliance
- Developing data contracts
- Certifying data sets
- Assessing and demonstrating value with KPIs
- DataOps and agile methodologies
- What we didn't say
  - Everything has to be a data product
  - You have to do it all at once
  - A data product must be a single, universal source of truth and there can be no siloes
  - You must be decentralized or operate a data mesh to have a data product

III. **Building External Data Products**
- Raising the bar
  - User expectations
  - ROI
  - Self-Service
  - Iterations
- Architecture considerations
  - Multi-tenancy
  - Native Snowflake Apps

IV. **Data Products IRL: Case Studies**
- Choozle
- Toast
- Fox

V. **Ship It!**

Appendix: Additional Resources



Chapter 1
Defining Data Products

I. Introduction

Data is no longer a second-class citizen. With better tooling, more advanced use cases, and a clearer understanding of its potential, many organizations have come to view their data ecosystem as a fully formed element of the company tech stack.

As a result, the most forward-thinking teams are adopting a new paradigm: treating data like a product. This guide will define the buzz and dive into exactly what it means to build a data product with concrete best practices and strategies from some of today's leading practitioners.

What does "data as a product" mean?

Data as a product or data products were most recently popularized as one of the four core principles of Zhamak Dehghani's data mesh paradigm, although academics and industry experts have been writing about their potential for several years.

We define data products to mean developing your critical data assets–such as key data sets, dashboards, or machine learning models–with the same reliable processes and architecture that you would use to develop a product for an external customer.

You can be expansive in the type of asset that is defined as a data product (is it an A/B testing platform, a multi-layered data platform, a data set, who cares?), but you must be exacting in the criteria (which we delve into in Chapter 2).

This all sounds straightforward in theory, but consider software-as-a-service (SaaS) solutions contractually guarantee uptimes in the range of " five 9s," or in other words 99.999%. Also, let's not forget the inherent resource and procedural challenges to developing a product.

Not only have product development processes evolved over decades to become incredibly sophisticated, but literal companies of people are formed to support them post-launch with entire departments organized around ensuring adoption, customer support, feature updates, and more.

But these challenges can also be an opportunity for savvy data teams. Here's how.

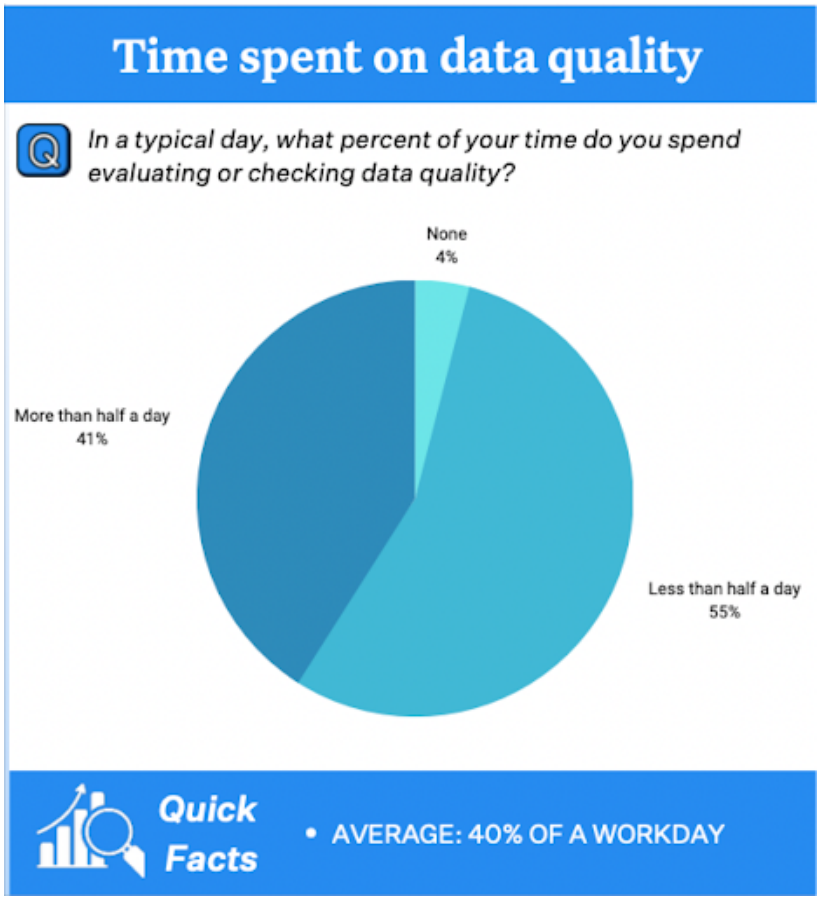
What are the benefits?

Data as a product is a lens and discipline that can help data teams create more scalable, reliable data systems. This comes with valuable benefits such as:

- Increased trust—Most business stakeholders will trust the data until they have a reason not to, but it only takes a few instances of conflicting or missing data to lose that benefit of the doubt. An increased focus on data quality ensures you retain data trust, which is important because, simply put, you cannot create a data-driven organization without it.

- Way to measure quantitative value of the data team —Too many data teams are evaluated based on a feeling. Stakeholders “feel like they are doing a good/bad job.” By treating data like a product you will be able to provide hard metrics to demonstrate how your team is performing and contributing to the business. That never hurts when asking for more resources.

- Improved focus and capacity —Treating data as a product is hard work and takes time. However, that investment pays dividends in the form of increased focus and capacity across your data engineering team. Believe it or not, there are multiple surveys (including one of our own) that reveal data engineers spend anywhere between a third and half of their time on data quality. The data product framework can help automate and expedite much of that tedious work so your team can spend less time fixing and more time innovating. Also, when your data consumers have effective self-service access to data products, the number of ad-hoc questions directed toward your team on Slack is reduced dramatically.



Survey of 300 data professionals commissioned by Monte Carlo suggests that - on average - data engineers spend 40% of their workday on data quality issues. Full survey.

Chapter 2
Designing a Data Product

II. Designing a Data Product

The concept of data product raises the bar for the availability, reliability, and usability of enterprise data, but it also represents a profound shift in perspective.

Waving a magic wand and declaring an executive dashboard a data product without any real changes to the data's underlying governance or reliability won't do your team - or customers - any favors.

Dashboard does not equal data product.

Creating SLAs

Data engineers need to measure data quality and data downtime just like our friends in software engineering who have addressed application downtime with specialized frameworks such as service level agreements, indicators, and objectives (SLA, SLI, SLOs).

Why go through the process of codifying SLAs at all, if you don't have a customer pressuring you to commit to certain thresholds in a contract? Why not just count on everyone to do their best and shoot for as close to 100% uptime as possible? Isn't this just introducing unnecessary red tape?

Not at all. The very practice of defining, agreeing upon, and measuring key attributes of what constitutes reliable software (or in our case data pipelines) can help engineering, product, and business teams align on what actually matters most and prioritize incoming requests.

With SLAs, different engineering teams and their stakeholders can be confident they're speaking the same language, caring about the same metrics, and sharing a commitment to clearly documented expectations.

Without these clearly defined metrics, consumers may make flawed assumptions or rely on anecdotal evidence about the reliability and trustworthiness of your data platform.

So how do you go about setting effective SLAs? For this, let's turn to the real world experience of Brandon Beidel, director of product management (data), at Red Ventures.

A data SLA dashboard. Image courtesy of Brandon Beidel.

His first step was to develop a solution for measuring data quality, which in his case was to leverage a data observability solution with automated monitoring, alerting, and reporting for data incidents. This is a critical step in setting realistic SLAs and improving on them–you can't improve what you can't measure.

He met with every business team in a weekly cadence, and without ever using the term "SLA," started having discussions around data quality and how it impacted their day to day.

The conversation was framed around simple business terms, how the data was being used, and the overall "who, what, when, where, and why." Specific questions included:

- How do you use this table?
- When do you look at this data? When do you report this data? Does this data need to be up to the minute, hourly, daily?
- What purpose does this serve?
- Who needs to get notified if this data is delayed?

He then created a SLA template for his data team to create and uphold SLAs across the business (the examples in the template below are ours) and started tracking performance of what percentage of SLAs were being met by each data warehouse in a dashboard.

Data SLA Template
Why **do you want this SLA?** What is the business need? Why does this SLA (Service Level Agreement) need to exist? Provide a brief description of the business need for this SLA. ***Example****: Company X reporting must be up to date in order to meet internal requirements on product performance reporting.*
Where **is the data?** Where are the specific data assets that are relevant to this SLA? List the tables, views or reports that are the most relevant to this SLA ***Example****: company_X.product_telemetry*
What **are the expectations for the data?** What do we need to check for? Describe the qualitative and quantitative attributes of the data we are checking for. ***Example:*** *We need to ensure that product telemetry data has been updated so the teams can take appropriate actions within each of their platforms AND so they can create any externally facing reports.*
When **does the data need to meet the expectations?** When does this become relevant? What specific deadlines matter for this SLA? Describe the days and times where the data needs to meet your expectations. Is it a recurring need? ***Example:*** *This is a daily need. Teams need to take action on yesterday's data by the beginning of the business day (i.e., 8:00 AM EST). Column X will never be null. Field X will always be equal to or greater than field Y. Table X will never decrease in size. No fields will be deleted on this table.*

***Who* is affected?**
Who needs to be notified if the expectations are not met?
Identify the specific teams and people that need to be informed, or who needs to take action.

***Example**: The people affected by this SLA are*
1. The Company X product teams, dedicated to building new product features.
2. Members of data team X assigned to support the product team

***How* will the team know if the SLA is met?**
How will the team identify if the expectations are not met?
Specifically describe the mechanism for monitoring this SLA. Is there a way to automate this?

***Example**: Monitoring and measurement will be done with Monte Carlo, and issues with be routed to #product-data-alerts Slack channel.*

***How* should the team respond if the expectations for data are not met?**
When an SLA is not met, what actions need to be taken? By whom? By when?
Describe the actions that need to be taken when this action isn't met.

***Example**: Within 2 hours after alert fires in the #product-data-alerts channel, a data team member assigned to the product team will:*

- *Verify if the SLA was in fact violated, or if the data is still delayed*
- *Communicate the status of the delay to the team in the #product-data channel*
- *If the issue persists, begin root cause analysis*

Within one business day
- *A ticket has been created*
- *The team has been updated on progress in #product-data*

Assigning Ownership

With the creation of data SLAs you now know who on the business side is using and impacted by your data. The next step is to create ownership and accountability within the data team at the organizational, project, and pipeline/table levels

Clear lines of ownership are important for obvious reasons. If everyone is accountable, then no one is actually accountable. It accelerates incident resolution and creates clear lines of communication.

Organizational Ownership

Let's start with creating accountability at the organizational level. So, who in your data organization owns the reliability piece of your data ecosystem?

As you can imagine, the answer isn't simple. From your company's CDO to your data engineers, it's ultimately everyone's responsibility to ensure data reliability. And although nearly every arm of every organization at every company relies on data, not every data team has the same structure, and various industries have different requirements.

For instance, it's the norm for financial institutions to hire entire teams of data governance experts, but at a small startup, not so much.

Below, we outline our approach to mapping data responsibilities, from accessibility to reliability, across your data organization using the RACI (Responsible, Accountable, Consulted, and Informed) matrix guidelines:

RACI Matrix for Data Personas						
	CDO	Business Intelligence	Data Science	Data Engineering	Data Governance	Product Manager
Facilitate Data Accessibility	A	R	C	R	C	R
Make it Easy to Interpret Data	A	R	R	C	I	C
Drive Insights & Recommendations Based on Data	A	R	R	C	C	C
Ensure Data Compliance	A	I	I	I	R	C
Maintain High Data Quality	A	C/I	R	R	C	R
Deliver on Data Reliability	A	C/I	C	R	I	R

At companies that ingest and transform terabytes of data (like Netflix or Uber), we've found that it's common for data engineers and data product managers to tackle the responsibility of monitoring and alerting for data reliability issues.

Barring these behemoths, the responsibility often falls on data engineers and product managers. They must balance the organization's demand for data with what can be provided reliably.

Notably, the brunt of any bad choices made here is often borne by the BI analysts, who's dashboards may wind up containing bad information or break from uncommunicated changes. In very early data organizations, these roles are often combined into a jack-of-all-trades data person or a product manager.

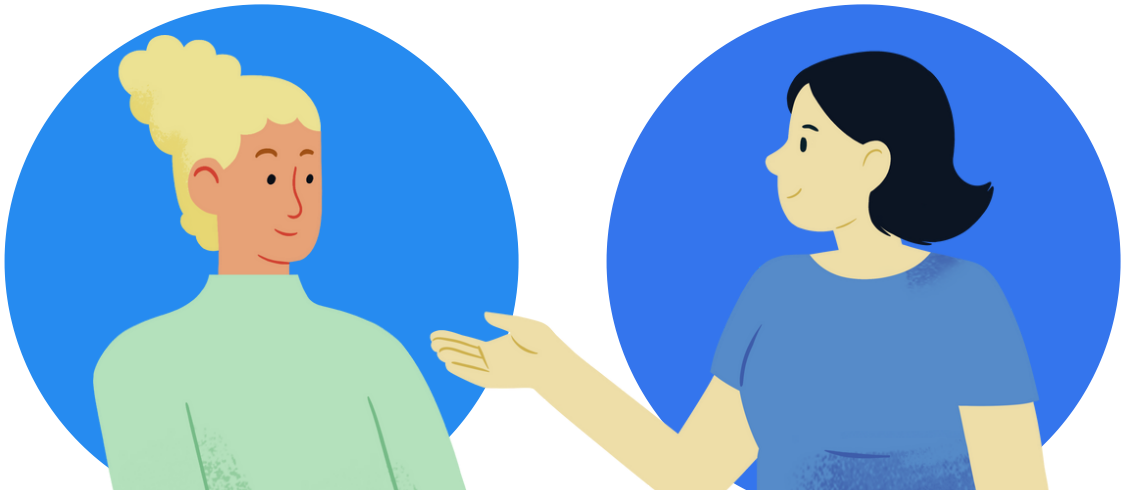
Project Ownership

Now, let's turn our attention to the project level. Just like every product will have a product manager, every data product should have a data product manager (one data product manager can oversee multiple products).

This is one of the most overlooked roles on the modern data team, but it's essential according to Wendy Turner-Williams, the Chief Data Officer at Tableau.

"From my perspective, a data product manager is actually one of the first roles I would hire for. I like them to actually create the vision and then drive the engineers to that vision," she said. "For me, it is a critical component as I tend to focus on data product managers who can create a story, engage with our internal customers or even our product team."

We also believe data product managers are critical to tackle tasks that build long-term value, but aren't in the purview of the typical day-to-day work of the data engineer. This includes acquiring user feedback, gathering team requirements, evaluating new technologies, building feature roadmaps, and more.



For example, in one of his former roles, Atul Gupte defined the product strategy and direction for Uber's data analytics, data knowledge, and data science platforms. Specifically, he led a project to improve the organization's data science workbench that was utilized by data scientists to make it easier to collaborate.

Data scientists were currently automating the process of validating and verifying worker documents that were required when applying to join the Uber platform. This was a great project for machine and deep learning, but the problem was data scientists would routinely hit limits of the available compute.

Whereas a traditional engineering project lead may have tried to add more virtual machines or extend the project timeline, Atul researched multiple solutions and identified virtual GPUs (then an emerging technology) as a possible solution.

While there was a high price tag, Atul justified the expenditure with leadership. The project was not only going to save the company millions, but supported a key competitive differentiator.

This proactive approach allowed Uber to start building the foundation they would need to leverage GPUs immediately upon availability. Time to value was greatly accelerated–a hallmark of a good data product manager.

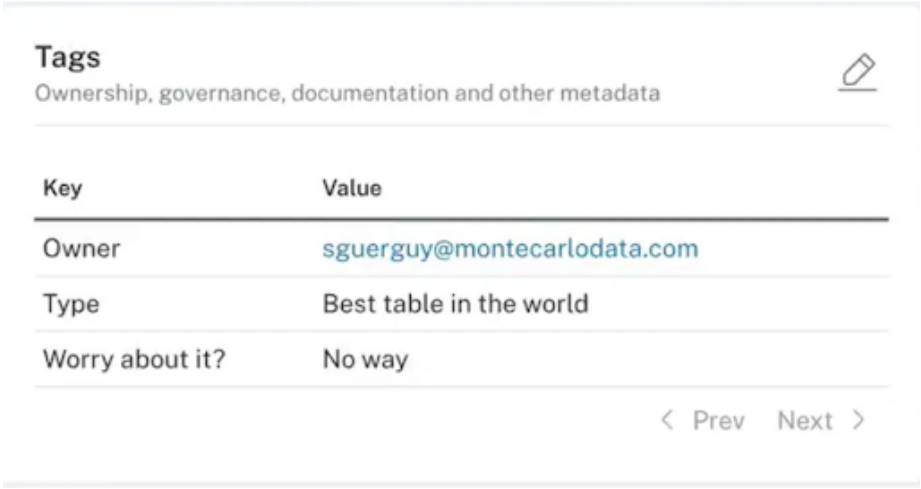
Pipeline/Table Ownership

As helpful as organizational and project level ownership is, when a data incident occurs there needs to be someone on the hook to coordinate the communication and triage/resolution process.

For most organizations, this is the data engineer who built or has the most experience with the pipeline or table that is experiencing the issue. That is a great system, but it needs to be transparent and well documented across the data team and with relevant stakeholders.

Key Assets						
Source	**Table Name**	**Important?**	**Importance Score**	**Avg. Reads Per Day**	**Total Users**	**...**
Redshift	jellyfish:cdw:salesforce.account	TRUE	1.00	7.56	8	...
Redshift	jellyfish:aquarium_ticket.sales:eb	TRUE	1.00	10.97	15	...
Redshift	jellyfish:analytics:looker.v3	TRUE	0.98	89.39	3	...
Redshift	jellyfish:ga:pageviews	FALSE	0.76	111.60	11	...

One of the simplest ways to do this is to identify your key tables (the ones that are populating your most important dashboards or that have the most read/writes) and label the owner. Data observability and data catalog solutions can both help identify key tables as well as serve as a reference point for ownership.



Monte Carlo allows owners to be assigned to tables along with other tags.

Speaking of documentation...

Documentation, Discovery, Self-Service

For a data asset to be considered a data product it needs to be well documented. That's because without good documentation it is difficult to enable discovery so people can find what they need because without good discovery, self-service just doesn't work.

And at that point, all ad-hoc questions start coming your way and instead of a data engineer, you become a data catalog. (You also need good documentation for governance, compliance, and quality use cases, and more on that in the subsequent sections, but here we're going to focus on documentation to enable self-service).

The challenge is your ability to pipe data is virtually limitless, but you are constrained by the capacity of humans to make it sustainably meaningful. In other words, you can never document all of your data, and rarely will you have as much documentation as you'd like.

So how can you solve this problem? In one word: focus. In many words:

- Fight data sprawl —More data doesn't simply translate into more or better decisions, in fact it can have the opposite effect. Ensure your data ecosystem prioritizes usability and organization. Some strategies include gathering data consumer needs upfront and only pipe that data, creating a consumer facing warehouse that is kept tidy by a layer of analytics engineers, or deprecating legacy data sets.

- Document key assets first —You can't document everything, so leverage the work you've done in the SLA and assigning ownership phases to start with the "key assets/tables" you've identified.

- Understand who needs self-service and for what use cases —It's rarely the case that self-service needs to be defined as every person in the company being able to answer every data set to answer every question they can think up without having to involve anyone in data engineering. A strong layer of data analysts fluent in writing SQL queries can help. Self-service also doesn't mean everyone across the organization should receive the same level of service. There are going to be some domains or departments that are more mature and require more self-service capabilities than others. Start with one team, make them deliriously happy, and scale from there.

- Iterate and automate —One company decided to hold off on a column level approach for their documentation since columns are often self-describing. Instead they focused on tables, views, and schemas for a "minimum viable product." They used their own system as well as Monte Carlo to identify key data assets and owners so they knew who to bother about documentation.

- Measure your documentation levels —The same company also implemented "stewardship analytics" that surfaced, for example, if 50% of the curated data is missing documentation. They then can have a conversation with that domain's steward and figure out how to remove blockers. Check out the scorecard below from former New York Times senior vice president of data and insights, Shane Murray. One of the key columns for measuring the maturation of their data products was the documentation level.

Illustrative – Table of Maturity Metrics

Maturity attributes are rated on a 5-point scale

		Scalable	Extensible	Reliable	Documented	Self-serve
Key Data Assets	Product Data	4	4	2	3	3
	Commerce Data	5	4	4	5	4
	Social Data	4	2	3	5	4
Data Management Infrastructure	ELT / Pipeline Tools	4	2	4	4	4
	Data Warehouse	5	5	5	2	3

- Leverage and integrate with other sources —There are multiple solutions inside and outside of the modern data stack that can contain helpful context on data assets. For example, one company has integrated documentation coming in from dbt within the Catalog plane of the Monte Carlo platform. Another solution is to integrate with or train data consumers to leverage what is typically an unused goldmine of information about a company's data: Slack. A quick Slack search for a table name before asking a question can save tons of time.

Governance and Compliance

To build a great data product, you need to factor in data compliance and data governance, two slightly different topics with equal importance.

Both activities involve documenting, classifying, and contextualizing the data in some manner, but we like to think of compliance as what you do for external stakeholders (regulators) to avoid fines and governance as what you do for your internal stakeholders (data consumers) to increase operational efficiencies. Let's dive into each.

Governance

Your data governance mission, should you choose to accept it, is to document the lineage, usage, compliance, business logic, quality, access, risk, and value of all data assets in the company along with our policies and processes.

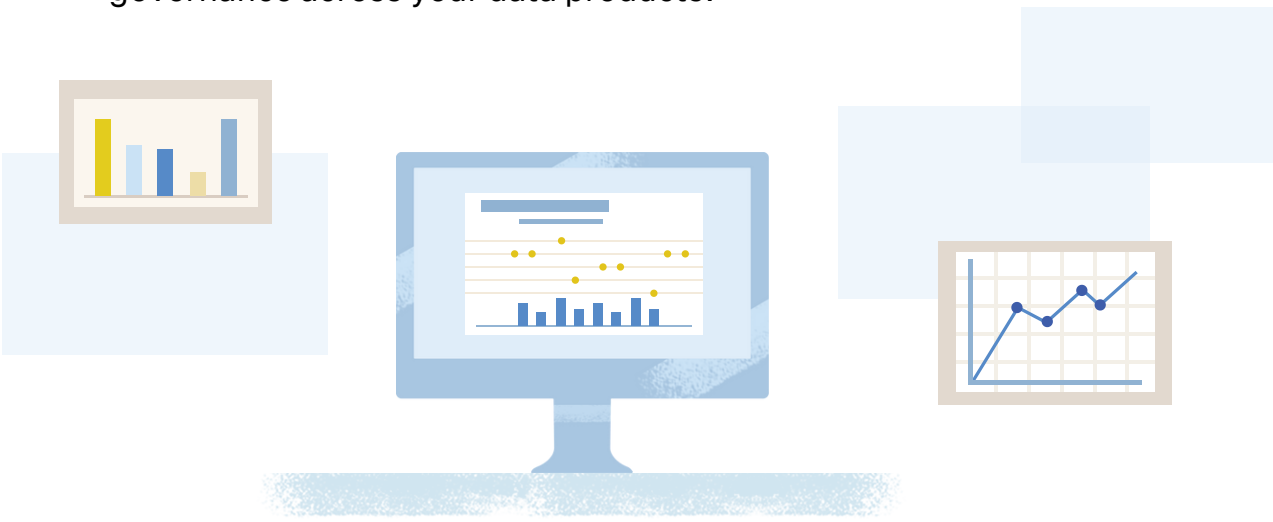
Yikes! Data governance can be a thankless task and has had more than its fair share of spins through the hype cycle.

It remains vital for data products, perhaps even more so as data volume levels rise. However, for these initiatives to succeed they need to be iterative, domain focused, and decentralized.

- Iterative —Too often, data governance somehow becomes translated as “data catalog.” And don’t get us wrong, data catalogs can be useful tools and have a roll to play, but efforts to catalog every data asset will inevitably fail. Focusing on the strategies listed above (SLAs, assigning owners, etc) for key assets will get you most of the way to where you want to go.

- Domain-focused —Data governance must also go beyond describing the data to understanding its purpose. How a producer of data might describe an asset would be very different from how a consumer of this data understands its function, and even between one consumer of data to another there might be a vast difference in terms of understanding the meaning ascribed to the data. A domain-first approach can give shared meaning to and requirements for the data within the operational workflow of the business. This is why the concepts of data product, data mesh and data governance complement one another so beautifully.

- Decentralized —We should recognize authority across the data team has begun to decentralize (dare we say, data mesh?) and decentralize the role of data governance as well. In other words, make sure someone in each domain has accountability for data governance across your data products.



- Clearcover Senior Data Engineering Manager Braun Reyes described how his organization has been successful deploying with a similar strategy.

"A one-size-fits-all, centralized governance approach did not work and was not going to scale. We have had much more momentum with a federated approach as detailed in the data mesh principles. Each data domain has a data steward that contributes to the data governance journey.

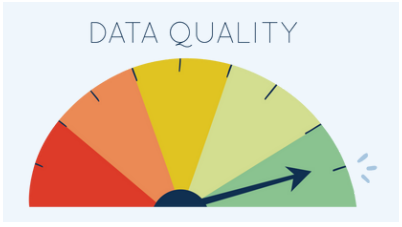
- Now, the proper incentives are in place. It all boils down to ownership. Governance had to be everyone's problem and it had to be easy to participate. Governance works best when each service that generates data is a domain with people who own the data and contract. It's their data, their enterprise relationship diagram (ERD), and their responsibility to document how to use it. We are still in the early stages, but starting to see real results and value."

Compliance

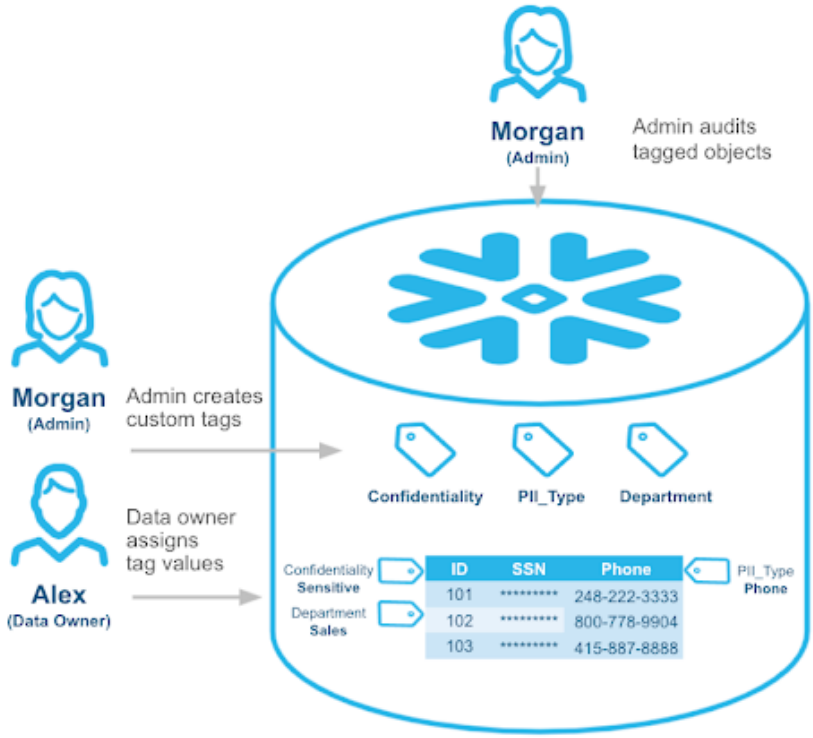
The bottom line for data products, and for data leaders in general, is that regulators require companies to say what they do, do what they say, and then prove it.

Understand the data regulations to which the data product is beholden (it often varies by country, industry, and even municipality) and document the compliance process. Most data regulations will focus on how personally identifiable information is acquired (chain of consent), secured/stored (more data sovereignty laws are passed every year), and that it is only accessible to those with a legitimate business need.

Acquire technology or tools—such as a data catalog, data loss prevention, privacy management platform, etc— if they are needed to enable or accelerate this work. In some cases, compliance for your data product can be as simple as a spreadsheet that documents which data assets contain PII and how it is being handled.



Many data warehouses and lakes will provide some functionality that can help with your compliance efforts. For example, Snowflake object tagging allows you to track sensitive data for compliance, discovery, protection, and resource usage. Available for Enterprise Edition and up, object tagging allows you to set up tags as key-value pairs that can indicate data classification or sensitivity.



Object Tagging in Snowflake. Source.

Having accurate object tagging in place makes it much easier to identify and monitor higher-risk data, and for compliance teams to implement additional security measures like dynamic data masking or row-level access restrictions.

Our customers in particular find this feature useful for tagging personally identifiable information (PII) data. For example, tagging a column with phone numbers as PII = “Phone Number.” Other helpful compliance features include Dynamic Data Masking and Row Access Policies.

Certifying Data Sets

Phew! We've done a lot of work so far to make our data product reliable and accessible. But how do we let our consumers know that the data sets powering them can be used with a higher degree of certainty to fuel more critical and valuable decisions and automations?

Data certification is your answer. Simply put, data certification is the process of taking tiered SLAs (many organizations use gold, silver, and bronze) and assigning them to data assets.

This helps with a few things. For one, as previously noted, it helps data consumers understand the level of trust they can have in the data. However, it also prevents data users from the facepalming mistake of leveraging the wrong asset, what we call the "You're using THAT table?!?" problem.

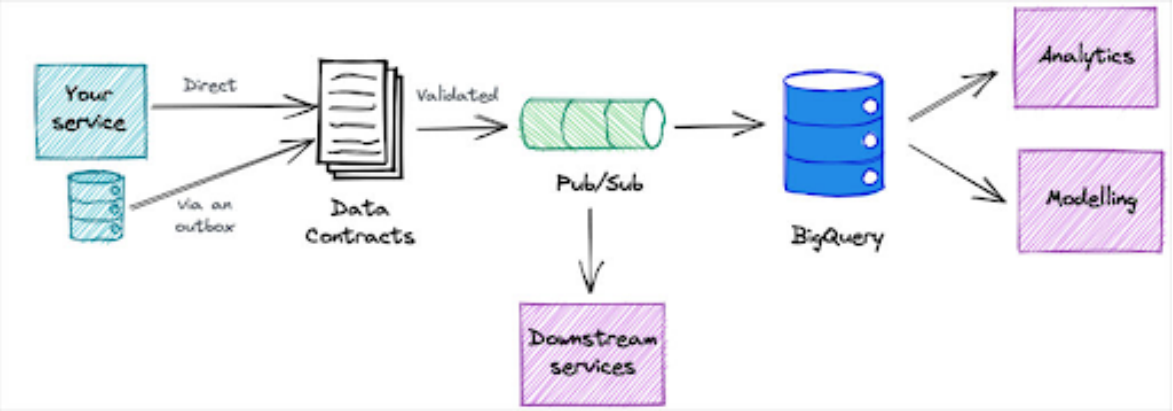
How are they supposed to know Michaels_data_warehouse2_beta is the right table? (Naming conventions by the way are a story for another time). Because you've certified it.

Finally, it can help with focus (notice a theme yet?). Treating every table or data asset with the same amount of rigor is a recipe for burnout and disaster.

Data Contracts

Data breaks bad in all sorts of ways, but one of the most frequent data quality issues arises when software engineers push service feature updates that change how data is produced. Unbeknownst to the software engineer, there is an entire data ecosystem leveraging that data and their new update has gummed up the works.

And how could they know? In most cases, software engineers and other upstream stakeholders do not have a comprehensive list of key data fields and how they are being used across the business. This is where data contracts come into play.

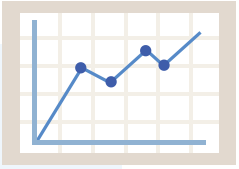


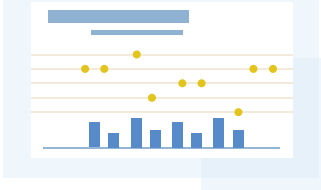
GoCardless' Data contract architecture. Image courtesy of Andrew Jones.

Data contracts are similar to an API in that they serve as documentation and version control to allow the consumer to rely on the service without fear it will change without warning and negatively impact their efforts.

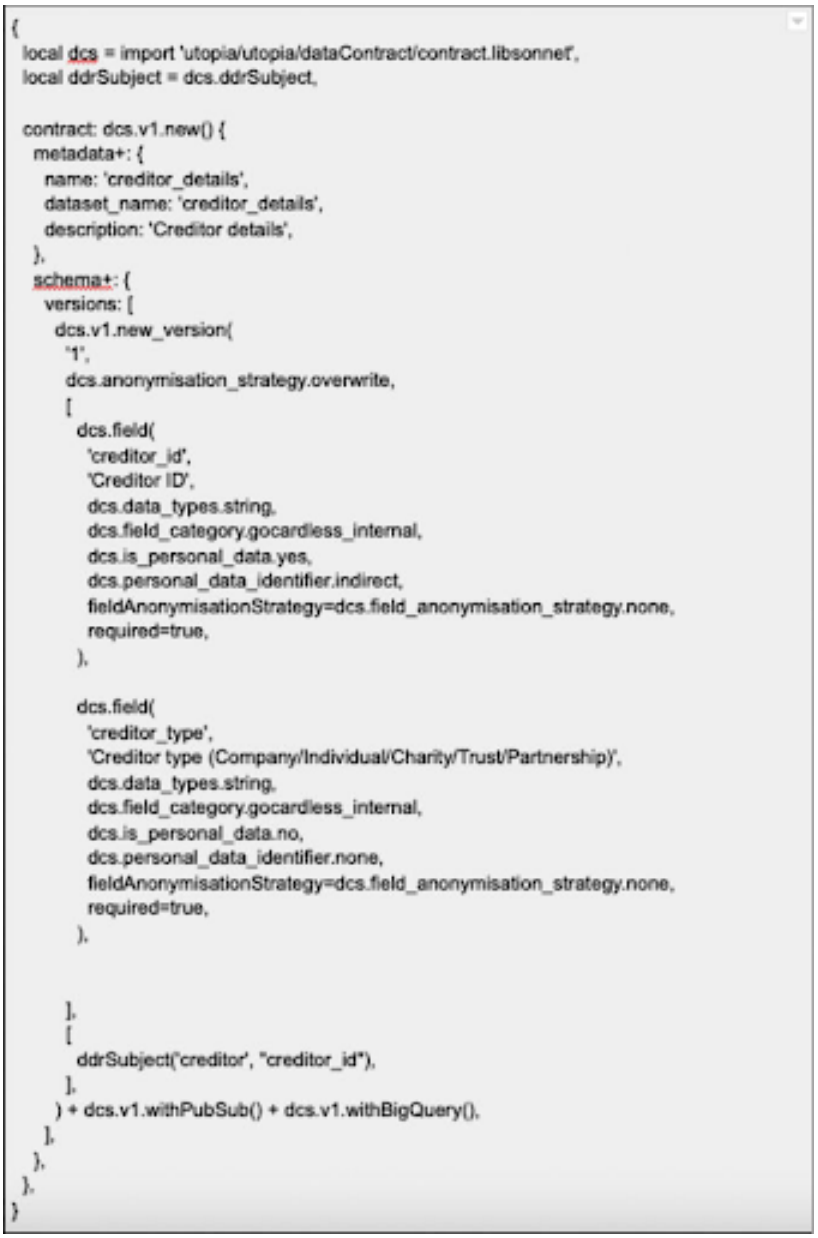
GoCardless has a self-service data contract model that is an interesting template for data teams to consider for their data products.

It starts with the data team using Jsonnet to define their schemas, categorize the data, and choose their service needs. Once the JSON file is merged in Github, dedicated BigQuery and PubSub resources are automatically deployed and populated with the requested data via a Kubernetes cluster.





For example, this is an example of one of our data contracts, which has been abridged to only show two fields.



```
{
 local dcs = import 'utopia/utopia/dataContract/contract.libsonnet',
 local ddrSubject = dcs.ddrSubject,

 contract: dcs.v1.new() {
  metadata+: {
   name: 'creditor_details',
   dataset_name: 'creditor_details',
   description: 'Creditor details',
  },
  schema+: {
   versions: [
    dcs.v1.new_version(
     '1',
     dcs.anonymisation_strategy.overwrite,
     [
      dcs.field(
       'creditor_id',
       'Creditor ID',
       dcs.data_types.string,
       dcs.field_category.gocardless_internal,
       dcs.is_personal_data.yes,
       dcs.personal_data_identifier.indirect,
       fieldAnonymisationStrategy=dcs.field_anonymisation_strategy.none,
       required=true,
      ),

      dcs.field(
       'creditor_type',
       'Creditor type (Company/Individual/Charity/Trust/Partnership)',
       dcs.data_types.string,
       dcs.field_category.gocardless_internal,
       dcs.is_personal_data.no,
       dcs.personal_data_identifier.none,
       fieldAnonymisationStrategy=dcs.field_anonymisation_strategy.none,
       required=true,
      ),


     ],
     [
      ddrSubject('creditor', "creditor_id"),
     ],
    ) + dcs.v1.withPubSub() + dcs.v1.withBigQuery(),
   ],
  },
 },
}
```

Assessing and demonstrating value with KPIs

When you are a startup, sometimes the focus is on customer usage and satisfaction rather than a hard ROI. Make customers happy and monetization will come later. It can be the same for small companies or young data teams—using adoption and data consumers as your north star for your data product will never leave you too far astray.

That being said, data teams need to hold themselves accountable to the same data-driven KPIs that their dashboards are driving for the rest of the company. The act of planning is valuable in itself, but demonstrating the value of a data product is important for obtaining and investing resources. Don't forget you wrote down to talk to your People team about hiring a data product manager after reading the ownership section of this chapter.

We recommend leveraging two core sets of KPIs. The first would be specific to the data and platform. These can measure:

- The breadth of customers and applications
- Building for the future
- Delivering business impact today
- Data quality

Let's start with the breadth of customers and applications. Apply the same level of rigor to your measurement of platform usage as you would a customer-facing product. Measure the footprint and speed of adoption, and run surveys to understand their satisfaction. While you should keep these surveys very short, it can also be an opportunity to gauge the drivers of their satisfaction, such as trust in the data / tools and ease of use. A scorecard of survey results might look like this:

Illustrative – Table of User Metrics

Survey questions are rated on a 5-point scale

		MAUs *– monthly active users*	**DAUs** *– daily active users*	**Ease of Use** *– how easily can you access the data you need for your job?*	**Trust** *– how trustworthy is the data provided by the tool(s)?*	**Satisfaction** *– overall, how satisfied are you with the tool(s)?*
User Applications	Business Intelligence	1200	220	3.0	4.0	2.0
	Experimentation	200	45	3.0	4.0	3.0
	Smart Hub / CDP	50	35	4.0	5.0	4.0

You also want to determine if your data product is falling behind, keeping pace with, or forging ahead of the business needs. A scorecard might look like this:

Illustrative – Table of Maturity Metrics

Maturity attributes are rated on a 5-point scale

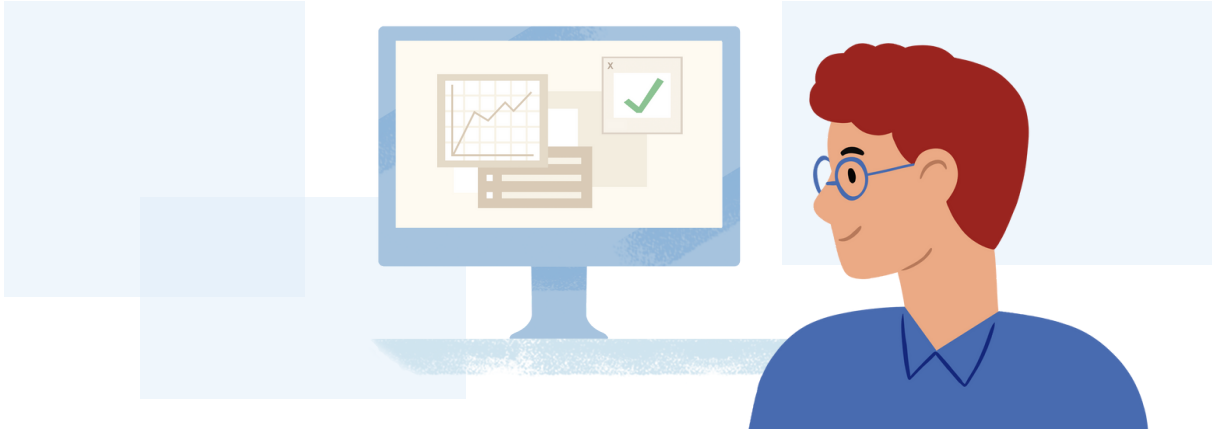
		Scalable	Extensible	Reliable	Documented	Self-serve
Key Data Assets	Product Data	4	4	2	3	3
	Commerce Data	5	4	4	5	4
	Social Data	4	2	3	5	4
Data Management Infrastructure	ELT / Pipeline Tools	4	2	4	4	4
	Data Warehouse	5	5	5	2	3

At the highest level, you must be able to quantify the impact driven by the data product, either in monetary terms ($, €, £) or a common unit of measurement that determines business success (e.g. sales, subscriptions). Precision is not the goal here – you can often tell if your platform bets are paying off by the order of magnitude of the value generated.

Value delivered is most visible from the applications to optimize product features or business processes. While it has become common-place for projects to measure the incremental lift in relative terms (e.g. 30% lift in conversion rate), you should ensure your teams are translating that lift into incremental value to the business. An example might look like this:

Illustrative – Table of Impact Metrics

		Population	Control Sales Rate	Relative Lift	Absolute Lift	Incremental Sales Value
Machine Learning Applications	Recommenders	5M	0.8%	1%	960	$96,000
	Ad Targeting	200K	5%	18%	720	$72,000



Finally, data quality is critical to creating a data-driven organization. The number of users your data product has doesn't mean much if they are being fed bad data or your dashboard is eternally being repaired. A good metric for this is data downtime, which is the number of incidents (N) multiplied by the time to detection (TTD) and time to resolution (TTR).

The data downtime equation.

The second set of KPIs should be shared goals, or how the data team has helped achieve business level objectives. For example, if you agree with engineering, product, and marketing that onboarding is a pain point, you can decide to build goals and KPIs around making it easier for new customers to get started.

By aligning the company around the shared goal of reducing new tool onboarding from five days to three days, for instance, you could begin to address the problem holistically: your data team gathers metrics on usage and helps build A/B tests, while your engineering team modifies the product, and your marketing team creates nurture campaigns. This is what it looks like to define and execute against a company-wide goal.

These goals also need to align with the culture of the business. For example, some organizations are going to be excited about how the data product has accelerated processes and others are going to be excited about how it has mitigated risk.

DataOps and Agile Methodologies

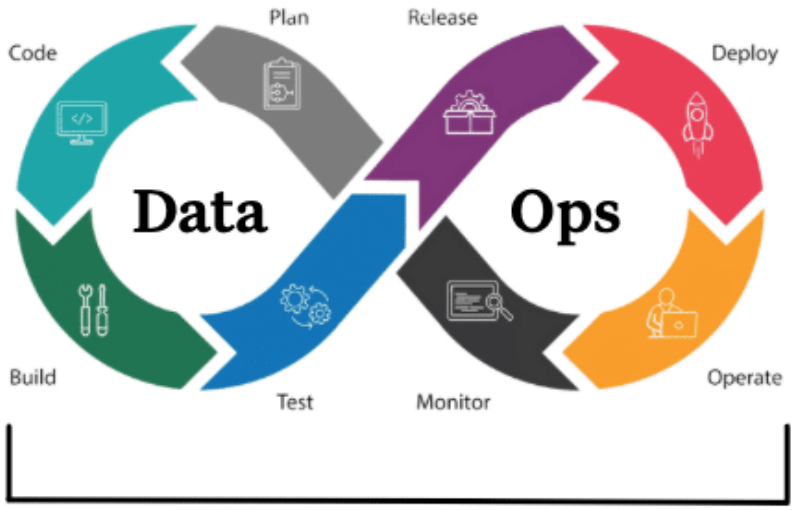
Everything we have discussed up to this point is how the data product should be built, but how it is built is important as well. This is where building data like a product, and borrowing proven processes from our friends in software development, can be especially helpful.

Before DevOps took the software engineering world by storm, developers were left in the dark once their applications were up and running.

Unfortunately, this led to the same mistakes occurring repeatedly as developers lacked insight into application performance and didn't know where to start looking to debug their code if something failed.

The solution? The now widely adopted concept of DevOps, a new approach that mandates collaboration and continuous iteration between developers (Dev) and operations (Ops) teams during the software deployment and development process.

DataOps is a discipline that merges data engineering and data science teams to support an organization's data needs, in a similar way to how DevOps helped scale software engineering.



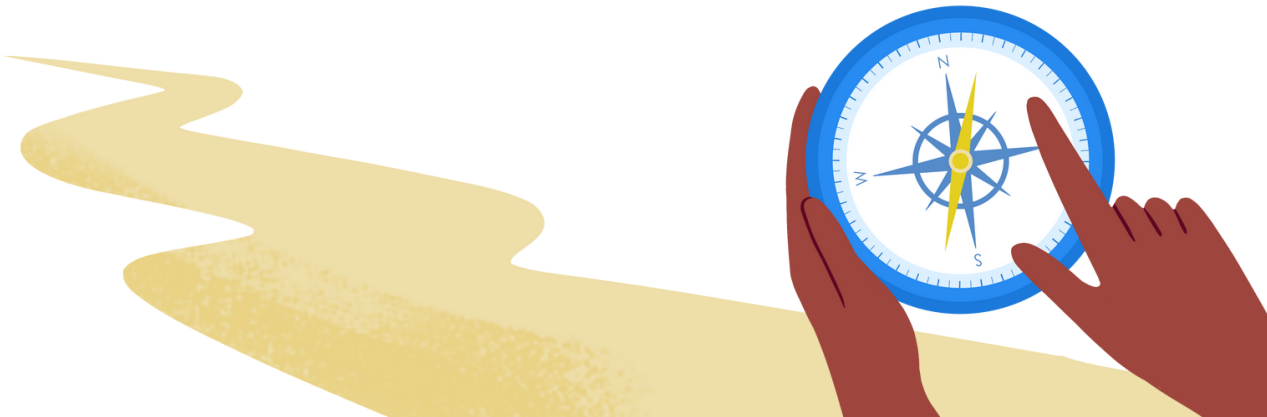
Similar to how DevOps applies CI/CD to software development and operations, DataOps entails a CI/CD-like, automation-first approach to building and scaling data products. At the same time, DataOps makes it easier for data engineering teams to provide analysts and other downstream stakeholders with reliable data to drive decision making.

Similar to how DevOps applies CI/CD to software development and operations, DataOps entails a CI/CD-like, automation-first approach to building and scaling data products. At the same time, DataOps makes it easier for data engineering teams to provide analysts and other downstream stakeholders with reliable data to drive decision making.

Here is what the DataOps lifecycle looks like in practice:

- Plan: Partnering with product, engineering, and business teams to set KPIs, SLAs, and SLIs for the quality and availability of data (more on this in the next section).
- Develop: Building the data products and machine learning models that will power your data application.
- Integrate: Integrating the code and/or data product within your existing tech and or data stack. (For example, you might integrate a dbt model with Airflow so the dbt module can automatically run.)
- Test: Testing your data to make sure it matches business logic and meets basic operational thresholds (such as uniqueness of your data or no null values).
- Release: Releasing your data into a test environment.
- Deploy: Merging your data into production.
- Operate: Running your data into applications such as Looker or Tableau dashboards and data loaders that feed machine learning models.
- Monitor: Continuously monitoring and alerting for any anomalies in the data.

Even if other team members are currently overseeing these functions, having a specialized role dedicated to architecting how the DataOps framework comes to life will increase accountability and streamline the process of adopting these best practices for data products.



What We Didn't Say

Data products are a relatively new concept and there are a lot of misconceptions around it. That's why it's important to clarify a few things we intentionally didn't specify because in our opinion they aren't necessary criteria for creating reliable, scaleable, well-adopted data systems.

Here are some things we didn't say:

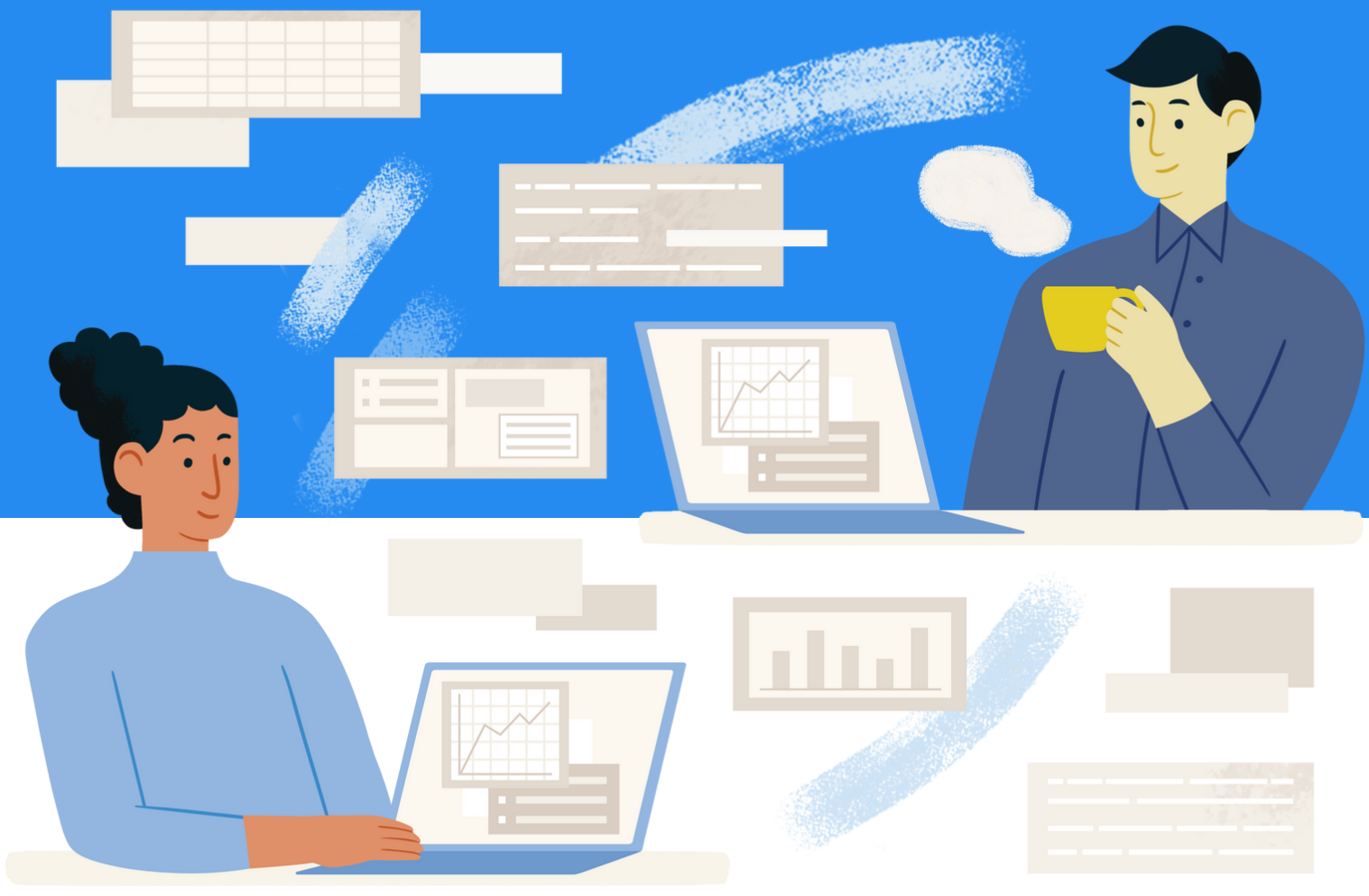
- Everything has to be a data product: This would be counterproductive. Sometimes you need to just spin up a quick dashboard. Also, when everything is a priority nothing is a priority. We are big believers in the value of focus and data products are a means to focus the team on your most important data assets.

- You have to do everything at once: This framework takes time and effort. Before you invest your precious resources into creating a data product, you may want to launch an initial version and see how well it is adopted or implement these processes in iterations and gather feedback.

- A data product must be a single, universal source of truth and there can be no siloes: Ideally, data is as accessible as possible, but the key for a data product is to enable specific teams to accomplish specific goals. Other data products in other domains might slice and dice things slightly differently for other data consumers, and that's OK.

- You must be decentralized or operate a data mesh to have a data product: While data products are one of the core principles of the data mesh, you can build a great data product with a centralized data team!

Chapter 3
Building External Data Products



III. External Data Products

An external data product is any data asset that either faces or impacts a customer. That can range from a dataset that is used in the customer billing process to a completely separate data intensive application with its own UI providing insights to a customer.

One of the hottest trends in data right now involves companies creating data applications or adding an additional layer within their SaaS product to help customers analyze their data. For example, a point of sale SaaS company might provide data back to its customers to better help it understand its sales trends.

However, an external data product doesn't necessarily need to be a fully built out application or integrate within the main SaaS offering. For example, Monte Carlo crosses the spectrum.

We are a data intensive SaaS application that monitors, alerts and provides lineage within our own UI. We also provide insights reports back to customers within our UI as well as providing them the option to surface it within their own Snowflake environment using the Snowflake data share integration.

In this latter case, we're just providing the building blocks for customers to be able to further customize how they'd like to visualize or combine it with other data.

Raising the Bar

Developing internal data products—whether a high-powered executive dashboard, a machine learning powered predictive buyer model for marketing, or a new customer model for the BI team—is still one of the most powerful ways data teams can add value to the organization.

But developing an external data product is a cut above: both in value added and in level of difficulty. It's a different motion that requires your team to build new muscle memory.

It’s a new way of thinking and requires elevated levels of coordination, discipline and rigor.
That isn’t to say it can’t be done by the same team, or that your internal data consumers can’t receive the same level of service as your external customers.

It’s important to have an expansive view of what is a data application or external data product because this should trigger the team to ensure it’s given a heightened level of rigor and it’s better to err on the side of over vs under engineering.

It’s important to ask:

- What external data products do we have and what types are they?
- Who are they serving? What are the use cases?
- Are they meeting those expectations? How are we measuring that?
- Do we have the right tools and processes in place?

Let’s dive into a few places where the bar is raised for external data products before focusing on perhaps the single most important differentiating factor–architecture.

User Expectations

Let’s say you have a data product that your user can usually trust to help answer some of their questions. The data is refreshed every day and the dashboard has some clickable elements where they can drill in for a more granular examination of the details.

That might be enough for some internal users. They can get their job done and their performance has even improved from when they didn’t have access to your slick dashboards.

On the other hand, your external users are pissed. They want to trust your product implicitly and have it answer all of their questions, all of the time, and in real-time.

And why shouldn't they be upset? After all, they are paying for your product and they could have gone with a competitor. When data is the product, data quality is product quality.

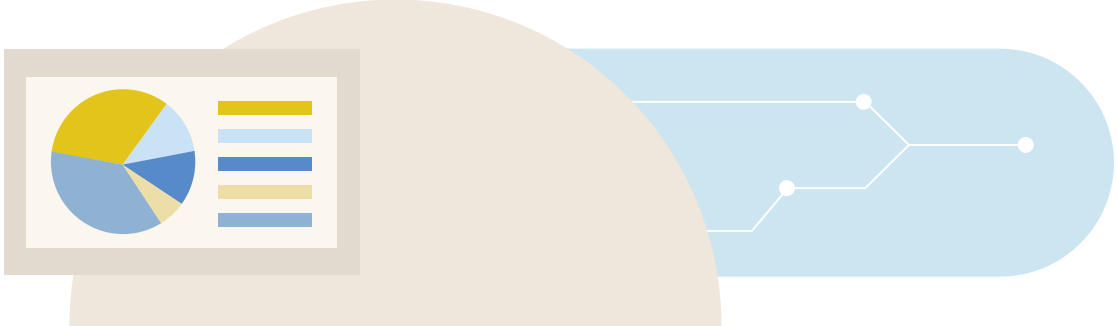
Yes Derek, data IS the product. No, the files aren't actually IN the computer.

This simple fact is why some of the most enthusiastic adopters of our data observability platform are leveraging it to support their data applications.

When data is customer facing or powering customer facing applications, poor quality can even break the product. For example, a data issue involving applications creating duplicate objects with the same primary key actually created an outage at Netflix.

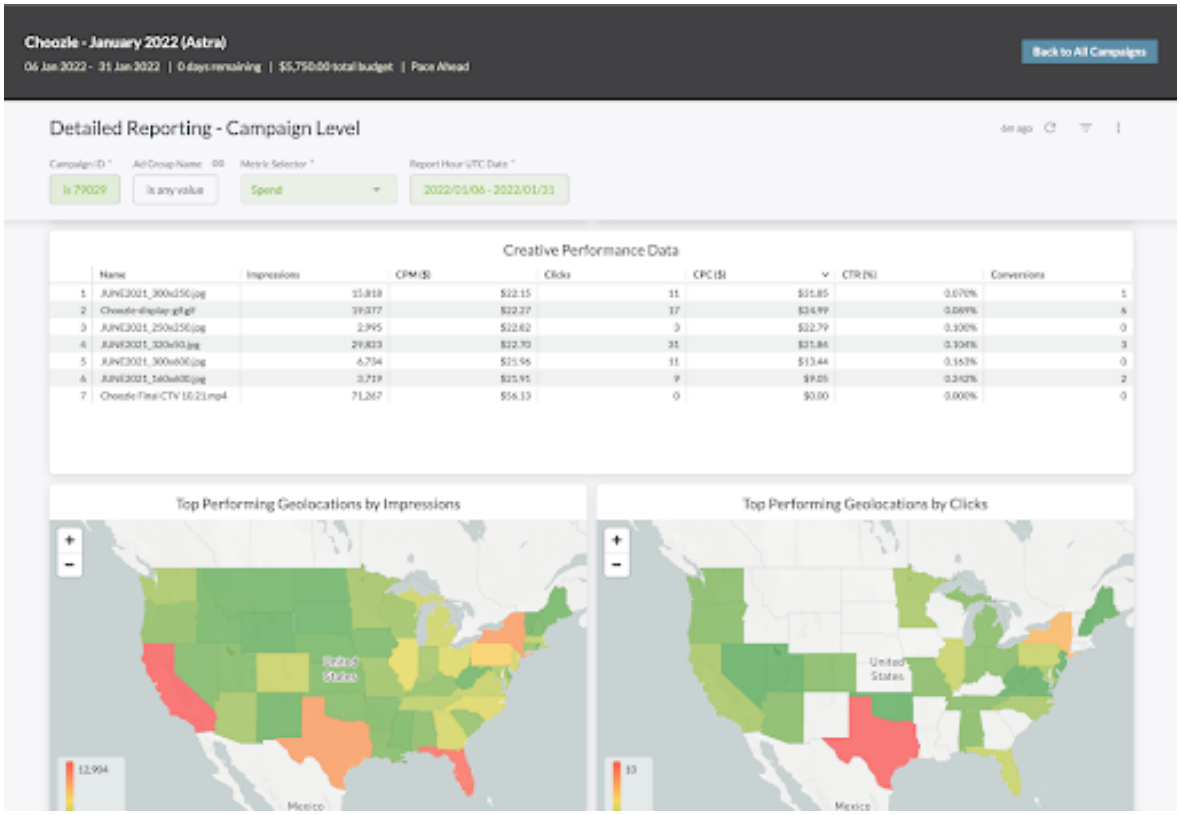
In regards to scale and speed, external customers never want to wait on the data and they want more data dimensions so they can slice and splice to their heart's content.

For example, one of our financial services customers has not just been focused on data freshness, but on data latency, or in other words the ability to load and update data in near real-time while also supporting queries.



Another customer, advertising platform Choozle, found additional data dimensions helpful for their upgraded platform:

“Snowflake gave us the ability to have all of the information available to our users. For example, we could show campaign performance across the top 20 zip codes and now advertisers can access data across all 30,000 zip codes in the US if they want it,” said Chief Customer Officer Adam Woods.



dimensions provided, are high.

ROI

The vast majority of data teams are not evaluated against a hard return on investment. While we advocate building KPIs and assessing value as part of your internal products, it is an absolute necessity to be capable of accurately measuring value when building an external data product. Product managers need to understand how to price it, and it must be profitable (at some point). They will need to know the start up costs for building the product as well as how much each component costs in providing the service (cost of goods).

This can be challenging for data teams that haven’t had the pleasure of building internal chargeback models for their data products that can differentiate, track, and charge customers according to scale of use.

Self-Service

“A-ha!” you say. “Our team already allows our internal consumers to self-service, this is nothing new.”

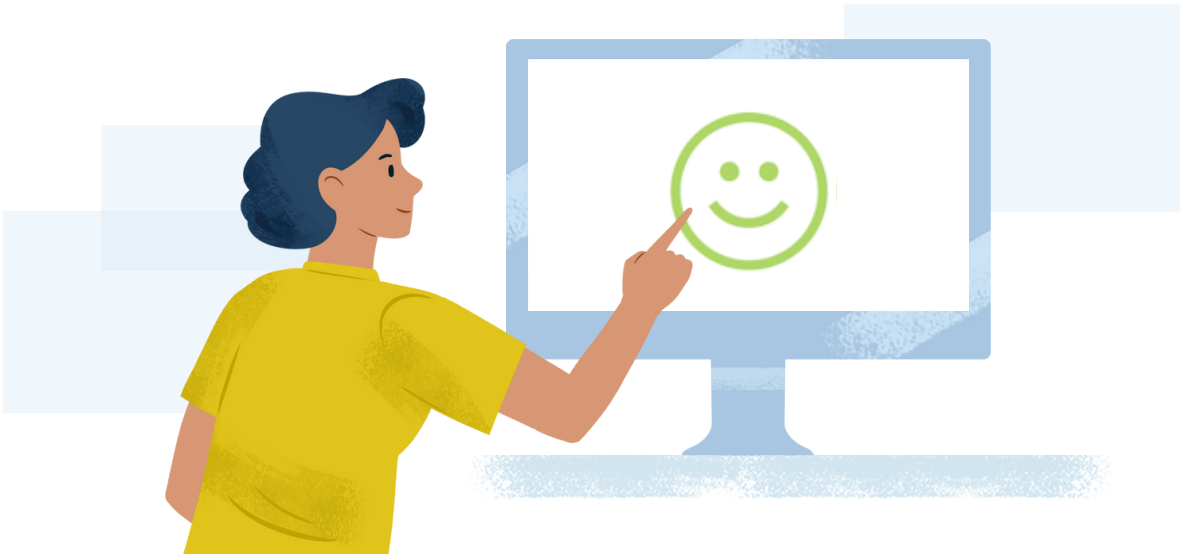
That may be true, but the bar has been raised for self-service and usability too.

Your external customers can’t Slack you to ask questions about the data or how you derived this customer’s likelihood to churn was, “3.5 out of 5 frowny faces.” The data product can’t be a black box–you need to show your work.

Is 3.5 frowny faces good? Why are they green, shouldn’t it be red? How is this calculated? I need more context from this hypothetical product!

How the user consumes and interfaces with your external data product requires a second thought as well. For many data teams the answer for their internal data product is, “...and then it’s surfaced in Looker.”

The UI has to be intuitive, the relevancy has to be immediate, and the context has to be evident.



Iterations

When you build your internal data products, it's often slow going at first as you gather requirements, build, and iterate with business stakeholders. After that, teams are often off and running to the next project. There will be patches and fixes for data downtime or maybe to meet internal SLAs if you're fancy, but on the whole you aren't refactoring those dashboards every quarter.

As previously mentioned, paying customers have higher expectations and they also have a lot more feedback. At Monte Carlo, it's not an exaggeration to say we are iterating every day. It's not all that rare of an occurrence to ship improvements based on customer feedback within days.

You need to know it's coming and build for it however. For example, Toast is extremely focused on the efficiency of their processes.

Not only do we listen to the business needs and obviously support them, but we also look internally and address scalability," said Toast data engineer Angie Delatorre. "If a job used to take one hour, and now it takes three hours, we always need to go back and look at those instances, so that shapes our OKRs as well.

When it comes to scaling operations, Snowflake director of product management Chris Child recommends,

"First, get all of your data in one place with the highest fidelity. Just get the raw data in there. Second, come up with repeatable pipelines for getting data to your analysts. You don't want to go back to the raw data every time you want to do something."

Former Uber data product manager Atul Gupte has discussed how critical it is when iterating data products to understand, "How to prioritize your product roadmap, as well as who you need to build for (often, the engineers) versus design for (the day-to-day platform users, including analysts)."

Architecture Considerations

External data products, like internal products, can leverage a wide variety of data cloud services to serve as the foundation of their platform including data lakes or warehouses.

Multi-Tenancy

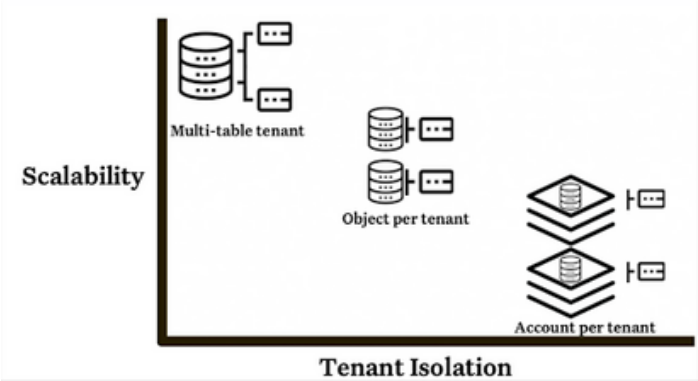
Many however will leverage a solution like Snowflake for its ability to optimize how relational data can be stored and queried at scale. Nothing new there.

What is new is that this will likely be your team's first discussion about multi-tenant architectures. That is a big change and decision point when serving external customers.

When leveraging a data warehouse as the product's foundation, there are three multi-tenant design options that Snowflake describes as:

- Multi-tenant table: Centralizes tenants in single, shared objects to enable tenants to share compute and other resources efficiently.
- Object per tenant: Isolates tenants into separate tables, schemas, databases, and warehouses within the same account.
- Account per tenant: Isolates tenants into separate Snowflake accounts.

Each option has advantages and disadvantages, but in an oversimplified nutshell, it depends on what needs to scale more efficiently–the shared compute/storage or role based access to data.



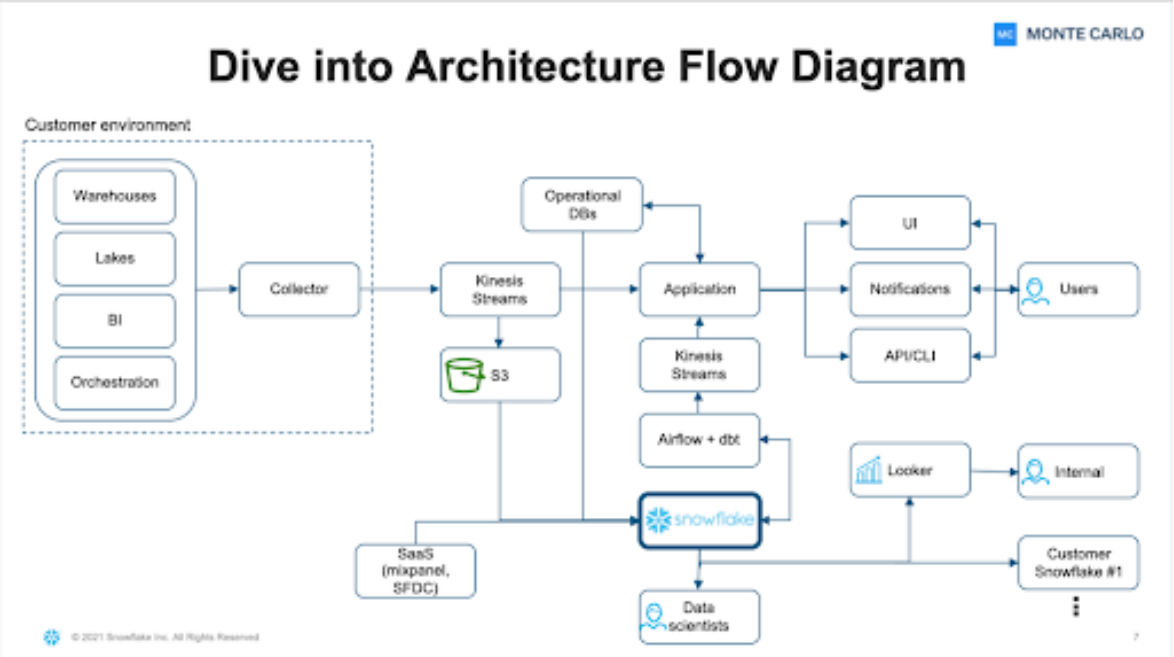
With a multi-tenant architecture the more isolation between tenants the more difficult it is to scale.

Most internal products are delivered within the same corporate umbrella with the same broad internal policies and regulations. The marketing team isn't going to be upset if its data assets are in the same warehouse as the legal team's for instance. External customers may care quite a bit more.

Of course, you can make other architecture choices across your stack to mitigate these trade-offs.

For example, Monte Carlo leverages a MTT multi-tenant architecture in Snowflake that logically separates customer data using industry best practices such as tokenization.

Additionally, we use a hybrid architecture with a data collector that is embedded within the customers' environment (often but not always as its own virtual private cloud).



That means data never leaves their environment. PII and sensitive data is abstracted away and what we pull are the non-sensitive logs, metric aggregations needed to assess their data systems' health.

Another part of the architecture decision making process, similar to internal data products, is understanding the use cases and workloads. What is the frequency, size, and required timeliness? Will customers receive data at a set time, be able to query data on-demand, access it in real-time, or all three?

As we have mentioned previously, understanding workloads is very helpful for making cost effective architectural choices. What is different with external products however, is there may be a wider variety of use cases to support.

In architecting Monte Carlo, we didn't just have to consider our mission-critical production workloads, but how our internal teams accessed this external-facing data as well. In this case, to conduct internal analytics and data science research as part of developing our machine learning powered anomaly monitors.

Native Snowflake Apps

At the 2022 Snowflake Summit, the company made a number of announcements that will enable data product developers to easily build native applications in Snowflake.

For example, Snowpark now supports Python and larger memory instances for machine learning. So for example, a developer can use Python in Snowpark to build out a predictive learning application for the marketing team to experiment with different advertising channels and how that will impact their ROI. They could then use Streamlit to easily build the application and visualization of their data science model for their consumers to leverage from those blocks of code before then publishing and monetizing it on the Snowflake Marketplace.

Leveraging Snowflake as your unified platform for developing data applications from code to monetization has a number of advantages. Developers don't have to worry about the massive roadblock of accessing customer data, because they can already access it securely via the Snowflake data share. Security, governance, distribution, serverless deployment, monetization–all of these challenges are addressed via the marketplace.

Snowflake native apps were announced at the 2022 Summit.

Chapter 4

Data Products IRL: Case Studies

IV. Data Products IRL: Case Studies

Choozle

Choozle is a leading digital advertising software company that gives small and medium-sized businesses access to enterprise grade advertising technology. You may remember them from Chapter 3.

Choozle's main data sources are demand-side platforms (DSPs), with which they facilitate the buying of media on behalf of its users. These data sources are static with defined ingest patterns, and are powered by ELT pipelines that bring data into a unified schema where Choozle abstracts the difference in how data is presented between the platforms.

That equation changed and a data quality issue arose when Choozle released its massively powerful unified reporting capability, which allows users to connect external media sources.

"When our advertisers connect to Google, Bing, Facebook, or another outside platform, Fivetran goes into the data warehouse and drops it into the reporting stack fully automated. I don't know when an advertiser has created a connector," said Choozle Chief Customer Officer Adam Woods. "This created table sprawl, proliferation, and fragmentation. We needed data monitoring and alerting to make sure all of these tables were synced and up-to-date, otherwise we would start hearing from customers."

Adam ruled out open source testing solutions due to the associated maintenance costs.

"I understand the instinct to turn to open source, but I actually have a lower cost of ownership with a tool like Monte Carlo because the management burden is so low and the ecosystem works so well together. After one phone call with the Monte Carlo team, we were connected to our data warehouse, and we had data observability a week later," said Adam.

“I love that with Snowflake and Monte Carlo my data stack is always up-to-date and I never have to apply a patch. We are able to reinvest the time developers and database analysts would have spent worrying about updates and infrastructure into building exceptional customer experiences.”

Monte Carlo gave the Choozle team deeper visibility into data product issues that otherwise may not have been proactively caught.

“Without a tool like this, we might have monitoring coverage on final resulting tables, but that can hide a lot of issues,” said Adam. “You might not see something pertaining to a small fraction of the tens of thousands campaigns in that table, but the advertiser running that campaign is going to see it. With Monte Carlo we are at a level where we don’t have to compromise. We can have alerting on all of our 3,500 tables.”

Every organization experiences some level of data downtime, or periods of time when data is partial, erroneous, missing or otherwise inaccurate. By leveraging Monte Carlo, Choozle has reduced their data downtime approximately 88%.

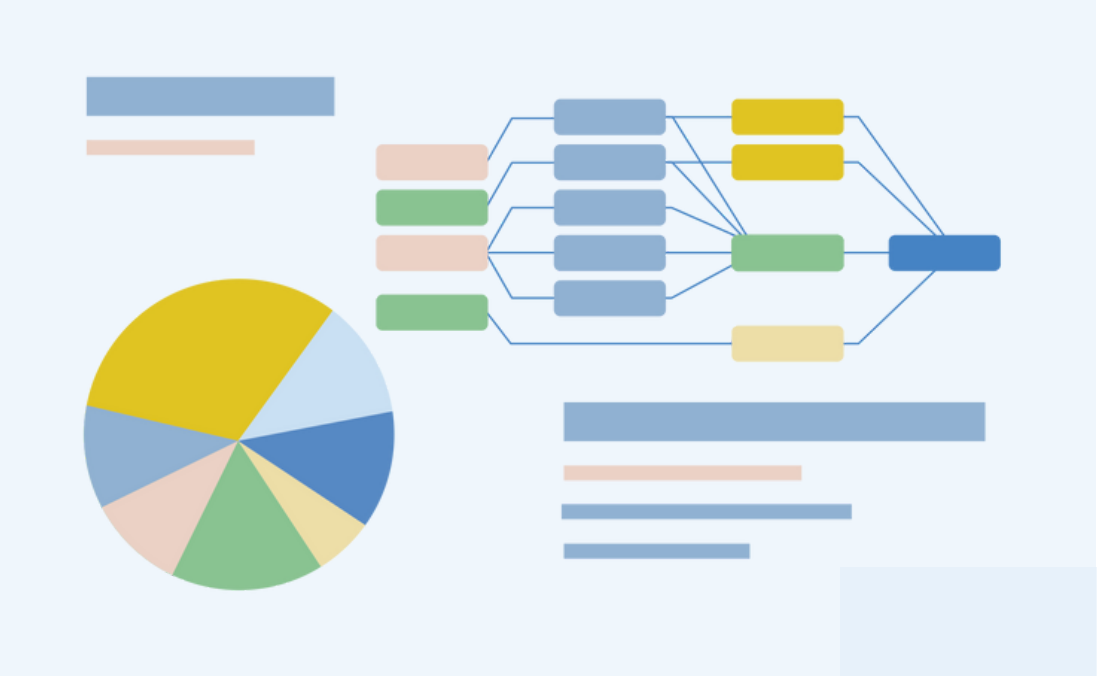
“We see about 2 to 3 real incidents every week of varying severity. Those issues are resolved in an hour whereas before it might take a full day,” said Adam. “When you are alerted closer to the time of the breakage it’s a quicker cognitive jump to understand what has changed in the environment.”

the Choozle team realized immediate results when they recently brought in a new stream of media into production. The team had missed a field containing the primary key that uniquely identifies each table record.

“We expect that field to not be null 100% of the time. We started getting a handful of campaigns, about .2%, where that value was null,” said Adam. “That led us to look at the situation and address it before anyone noticed there were certain campaigns with certain views not seeing any data. That type of problem could have magnified over time and filled tables with all kinds of junk. With Monte Carlo, our time-to-detection in this case was accelerated from days to minutes.”

Adam's advice for other data professionals is to be bold: the risk of innovation has become greatly mitigated over time with the rise of cloud data warehouses like Snowflake and SaaS platforms like Monte Carlo.

"A decade ago when I was running warehouses on-prem, it was very hard to innovate. Every answer to a new suggestion was a no because you don't want to break the whole warehouse. I encourage my developers to be more aggressive. You aren't going to break Snowflake by writing a bad query, and if you did we would know it right away," said Adam. "It's the same with Monte Carlo. If you already have Snowflake, there is almost no risk in trying. The cost to get set up was the salary for three people for an hour and we saw value immediately."



Toast

Toast is a leading point of sale provider for restaurants and a recent unicorn with a thoughtful data team focused on building out their data platform.

The Toast data team began showing how their data platform and products provide value to the business by first understanding the business problems affecting their colleagues.

“Our director gets our team involved early into these problems and helps us understand how we’re going to solve it, how we’re going to measure that we’re solving it correctly, and how we’re going to start to track that data—not just now but also in the future,” said Noah Abramson, Data Engineering Manager, Toast.

Over time, Toast developed a system that removed bottlenecks by building a self-serve analytics model to service the broader company and remove bottlenecks.

“[Our process was originally] super centralized, and we owned the entire stack,” explained Noah. “As the company started to grow, it got overwhelming. We pivoted to a self-service model, and we became the team that you would consult with as you were building these dashboards and owning the data.”

At Toast, the data platform team owns the company’s external-facing data insights and analytics.

“One of our big value ads [as an organization] is giving business insights to our customers: restaurants,” says Noah. “How did they do over time? How much were their sales yesterday? Who is their top customer? It’s the data platform team’s job to engage with our restaurant customers.”

Noah’s team, in contrast, is largely internal.

“We say our customers are all Toast employees,” he says. “We try to enable all of them with as much data as possible. Our team services all internal data requests from product to go-to-market to customer support to hardware operations.”

It's thus Noah's team's job to build out data flows into overarching systems and help stakeholders across the organization derive insights from tools including Snowflake and Looker.

When building data products, it's important both to measure how stakeholders can leverage data to support business needs and to ascertain the quality and efficiency of the data team's performance.

"We really listen to what the business needs," says Noah regarding how his team thinks about measuring data-related KPIs. "At the top level, they come up with a couple different objectives to hit on: e.g., growing customers, growing revenue, cutting costs in some spend area."

Noah and his team then take these high-level business objectives and use them to build out Objectives and Key Results (OKRs).

We're able to do that in a few ways," says Noah. "If you think about growing the customer base, for example, we ask, 'how do we enable people with data to make more decisions?' If somebody has a new product idea, how do we play with that and let them put it out there and then measure it?"

Additionally, the team focuses on measuring the scalability of its processes. "Not only do we listen to the business needs and obviously support them, but we also look internally and address scalability," says Angie Delatorre, Data Engineer, Toast. "If a job used to take one hour, and now it takes three hours, we always need to go back and look at those instances, so that shapes our OKRs as well."

As the Toast team began working with ever-growing volumes of data, ensuring data reliability became mission-critical.

"There's a lot of moving parts," says Noah of the data stack. "There's a lot of logic in the staging areas and lots of things that happen. So that kind of begs the question, how do we observe all of this data? And how do we make sure when data gets to production and Looker that it is what we want it to be, and it's accurate, and it's timely, and all of those fun things that we actually care about?"

Originally, Noah and two other engineers spent a day building a data freshness tool they called Breadbox. The tool could conduct basic data observability tasks including storing raw counts, storing percent nulls, ensuring data would land in the data lake when expected, and more.

"That was really cool," notes Noah, "but as our data grew, we didn't keep up. As all of these new sources came in and demanded a different type of observation, we were spending time building the integration into the tool and not as much time in building out the new test for that tool."

Once the team reached that pivotal level of growth, it was time to consider purchasing a data observability platform rather than pouring time and resources into perfecting its own.

"With Monte Carlo, we got the thing up and running within a few hours and then let it go," says Noah. "We've been comparing our custom tool to Monte Carlo in this implementation process. We didn't write any code. We didn't do anything except click a few buttons. And it's giving us insights that we spent time writing and building or maintaining."

Fox

Fox is more than just a news, media and sports powerhouse; they're also behind one of the media industry's most advanced data architectures. Here's how the early adopters of AWS Redshift, Kinesis and Apache Spark are democratizing data across the organization with data observability in mind.

Companies are ingesting and storing an incredible amount of data—but not every organization knows how to realize its full value.

Data often gets stuck in silos, with requests backing up in ticket queues that never reach overworked data engineers and analysts struggling to serve the needs of their entire organization.

As VP of Data Services at media giant Fox Networks, Alex Tverdohleb has spent the last several years focusing on this problem. He intentionally built out the teams, the technology, and the trust necessary to give internal stakeholders across the digital organization the freedom to conduct ad-hoc analytics without getting centralized data engineers and analysts involvement as much as possible.

As distributed architectures continue to become a new gold standard for data driven organizations, this kind of self-serve motion would be a dream come true for many data leaders. So when the Monte Carlo team got the chance to sit down with Alex, we took a deep dive into how he made it happen.

Here's how his team architected a hybrid data architecture that prioritizes democratization and access, while ensuring reliability and trust at every turn.

Exercise "Controlled Freedom" When Dealing With Stakeholders

Alex has built decentralized access to data at Fox on a foundation he calls "controlled freedom." In fact, he believes using your data team as the single source of truth within an organization actually creates the biggest silo.

So instead of becoming a guardian and bottleneck, Alex and his data team focus on setting certain parameters around how data is ingested and supplied to stakeholders. Within the framework, internal data consumers at Fox have the freedom to create and use data products as needed to meet their business goals.

“If you think about a centralized data reporting structure, where you used to come in, open a ticket, and wait for your turn, by the time you get an answer, it’s often too late,” Alex said. “Businesses are evolving and growing at a pace I’ve never seen before, and decisions are being made at a blazing speed. You have to have data at your fingertips to make the correct decision.”

To accomplish this at scale, Alex and his centralized data team control a few key areas: how data is ingested, how data is kept secure, and how data is optimized in the best format to be then published to standard executive reports. When his team can ensure data sources are trustworthy, data is secure, and the company is using consistent metrics and definitions for high-level reporting, it gives data consumers the confidence to freely access and leverage data within that framework.

“Everything else, especially within data discovery and your ad-hoc analytics, should be free,” said Alex. “We give you the source of the data and guarantee it’s trustworthy. We know that we’re watching those pipelines multiple times every day, and we know that the data inside can be used for X, Y, and Z — so just go ahead and use it how you want. I believe this is the way forward: “striving towards giving people trust in the data platforms while supplying them with the tools and skill sets they need to be self-sufficient.”

Invest in a Decentralized Data Team

Under Alex’s leadership, five teams oversee data for the Fox digital organization: data tagging and collections, data engineering, data analytics, data science, and data architecture. Each team has its own responsibilities, but everyone works together to solve problems for the entire business.

“I strongly believe in the fact that you have to engage the team in the decision-making process and have a collaborative approach,” said Alex. “We don’t have a single person leading architecture—it’s a team chapter approach. The power of the company is, in essence, the data. But people are the power of that data. People are what makes that data available.”

While members of different data teams collaborate to deliver value to the business, there’s a clear delineation between analysts and engineers within the Fox data organization. Analysts sit close to the business units, understanding pain points and working to find and validate new data sources. This knowledge informs what Alex and his teams call an STM, or Source to Target Mapping—a spec that essentially allows engineers to operate from a well-defined playbook to build the pipelines and architecture necessary to support the data needs of the business.

This division of labor between analysts and engineers “allows people to focus on their specific areas instead of being spread thin,” said Alex. “Some people may disagree with me, but quite frankly, having developers attend a lot of business meetings can be a waste of their time—because collecting and understanding business requirements often is a strenuous and time consuming effort. By installing the analytics before engineering gets involved, we can bridge that gap and then allow the developers to do what they do best – building the most reliable, resilient and optimized jobs .”

(It’s worth noting, however, that this decentralized approach won’t work for every organization, and the needs of your team structure will vary based on the SLAs your company sets for data.)

Avoid Shiny New Toys in Favor of Problem-Solving Tech

Under Alex’s leadership, five teams oversee data for the Fox digital organization: data tagging and collections, data engineering, data analytics, data science, and data architecture. Each team has its own responsibilities, but everyone works together to solve problems for the entire business.

“I strongly believe in the fact that you have to engage the team in the decision-making process and have a collaborative approach,” said Alex.

“We don’t have a single person leading architecture—it’s a team chapter approach. The power of the company is, in essence, the data. But people are the power of that data. People are what makes that data available.”

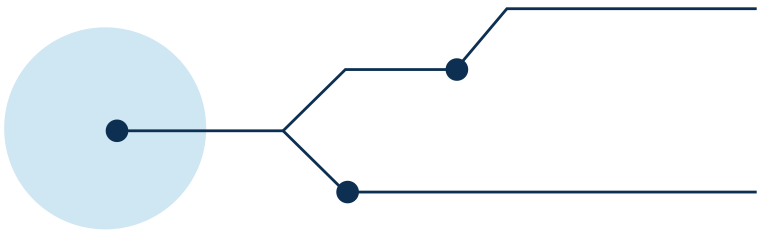
While members of different data teams collaborate to deliver value to the business, there’s a clear delineation between analysts and engineers within the Fox data organization. Analysts sit close to the business units, understanding pain points and working to find and validate new data sources. This knowledge informs what Alex and his teams call an STM, or Source to Target Mapping—a spec that essentially allows engineers to operate from a well-defined playbook to build the pipelines and architecture necessary to support the data needs of the business.

This division of labor between analysts and engineers “allows people to focus on their specific areas instead of being spread thin,” said Alex. “Some people may disagree with me, but quite frankly, having developers attend a lot of business meetings can be a waste of their time—because collecting and understanding business requirements often is a strenuous and time consuming effort. By installing the analytics before engineering gets involved, we can bridge that gap and then allow the developers to do what they do best – building the most reliable, resilient and optimized jobs .”

(It’s worth noting, however, that this decentralized approach won’t work for every organization, and the needs of your team structure will vary based on the SLAs your company sets for data.)

I’ve been in data for over a decade, and I can say under no uncertain terms that Fox has one of the most robust and elegant data tech stacks that I’ve ever seen. But Alex is adamant that data leaders shouldn’t pursue shiny new tech for its own sake.

“First and foremost, in order to be successful at delivering the right underlying architecture, you need to understand the business,” said Alex. “Don’t chase the latest and greatest technology, because then you’re never going to stop. And sometimes the stack you have right now is good enough—all you have to do is optimize it.”



“First and foremost, in order to be successful at delivering the right underlying architecture, you need to understand the business,” said Alex. “Don’t chase the latest and greatest technology, because then you’re never going to stop. And sometimes the stack you have right now is good enough—all you have to do is optimize it.”

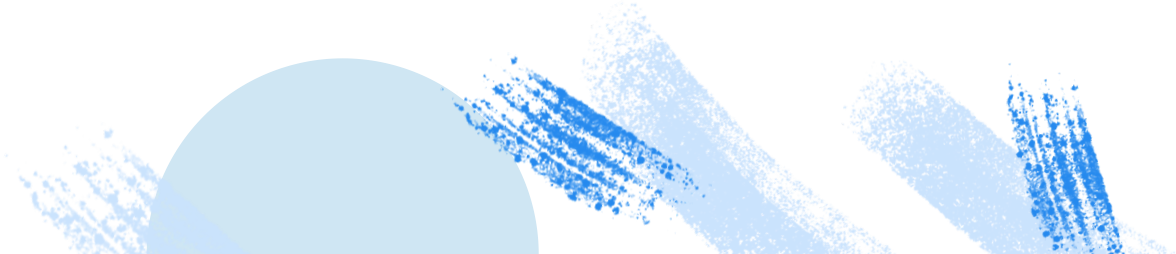
The Fox data team built their tech stack to meet a specific need: enabling self-service analytics. “We embarked on the journey of adopting a lakehouse architecture because it would give us both the beauty and control of a data lake, as well as the cleanliness and structure of a data warehouse.”

Several types of data flow into the Fox digital ecosystem, including batched, micro-batched, streaming, structured, and unstructured. After ingestion, data goes through what Alex refers to as a “three-layer cake”.

“First, we have the data exposed at its raw state, exactly how we ingest it,” said Alex. “But that raw data is often not usable for people who want to do discovery and exploration. That’s why we’re building the optimized layer, where data gets sorted, sliced-and-diced, and optimized in different file formats for the speed of reading, writing, and usability. After that, when we know something needs to be defined as a data model or included in a data set, we engage in that within the publishing layer and then build it out for broader consumption within the company. Inside of the published layer, data can be exposed via our tool stack.”

The optimized layer makes up the pool of data that Alex and his team provide to internal stakeholders under the “controlled freedom” model. With self-serve analytics, data users can discover and work with data assets that they already know are trustworthy and secure.

“If you don’t approach your data from the angle that it’s easy to discover, easy to search, and easy to observe, it becomes more like a swamp,” said Alex. “We need to instill and enforce some formats and strict regulations to make sure the data is getting properly indexed and properly stored so that people can find and make sense of the data.”



To Make Analytics Self-Serve, Invest in Data Trust

For this self-serve model to work, the organization needs to have trust that the data is accurate, reliable, and trustworthy. To help achieve this goal, the entire data stack is wrapped in QA, validation, and alerting. Fox uses Monte Carlo to provide end-to-end data observability, along with Datadog, Cloud Watch Alerts, and custom frameworks to help govern and secure data throughout its lifecycle.

“Data observability has become a necessity, not a luxury, for us,” said Alex. “As the business has become more and more data-driven, nothing is worse than allowing leadership to make a decision based upon data that you don’t have trust in. That has tremendous costs and repercussions.”

Alex estimates that the Fox digital organization receives data multiple times a day from over 200 sources. They process nearly 10,000 schemas and tens of billions of records per week. “You can’t scale the team to maintain and support and validate and observe that amount of data. You have to have at least a few tools at your disposal. For us to make sure that we have trust in the data’s timeliness, completeness, and cleanliness, tools like Monte Carlo are “must-to-have” . It’s been a great addition to allow us to build an AI-powered overview of what’s happening in our data stack.”

The continual monitoring and alerting Monte Carlo provides, along with automated data lineage, helps Alex’s team to be more proactive about data incidents when they do occur. “We can catch the issues before they hit production and if they do, we know the level of impact by using reverse-engineering to see how many and what kind of objects have been involved, and we can stop it in-flight before it causes a massive impact downstream. It all comes with trust—the moment you drop transparency or start hiding things, people lose trust and it’s really hard to regain it back. I’ve learned that no matter what happens, if you’re being honest and you’re owning the problem, people tend to understand and give you another chance to fix it.”

With the right tech, the right people, and the right processes in place, Alex and his teams have earned the trust required to build a self-serve data platform that powers decisions on a daily basis.

I don’t know about you, but I’m sold.

Chapter 5
Shipping a Data Product

V. Ship It!

While this guide may have read like a lot of work and a list of reasons why you shouldn't build a data product, we hope it helps demystify the challenges associated with this daunting but worthwhile endeavor.

You likely will not build the perfect data product on your first sprint (few do), but we encourage you to build, ship, iterate, rinse, and repeat.

You also don't have to undergo this ordeal alone. If you are interested in improving data quality and building awesome data products your company will actually use, talk to us. We've navigated this journey with hundreds of data teams and can help ensure you reach your final destination.

Additional Resources

Don’t let this be the ending point of your data quality journey! Check out more helpful resources including:

- Data Downtime Blog: Get fresh tips, how-tos, and expert advice on all things data.
- O’Reilly Data Quality Framework: The first several chapters of this practitioner's guide to building more trustworthy pipelines are free to access.
- Data Observability Product Tour: Check out this video tour showing just how a data observability platform works.
- Data Quality Value Calculator: Enter in a few specifics about your data environment and see how much you can save with data observability.
- Request A Demo: Talk to our team to get a more accurate assessment of your data downtime, its costs, and what level of value you can expect from Monte Carlo.