Data Contracts 101

andrew@andrew-jones.com

https://andrew-jones.com

Introduction

Things are changing fast in data.

Organization leaders are expecting to do more with data. They want to use their data as a competitive advantage, creating revenue-driving applications and taking advantage of the latest advancements in AI.

In fact, a survey from Nash Squared found that 64% of organizations think that big data and analytics are the way to deliver competitive advantage. Yet only 1 in 5 are using it to deliver increased revenue.

But on the other hand, data teams are struggling to deliver that value, due to poor data quality.

They are spending far too much of their time fixing their data pipelines, with failures often caused by upstream changes outside of their control.

57% of data practitioners highlight data quality as one of their chief obstacles in preparing data for analysis, according to a survey from dbt Labs. That's up from 41% the year before.

If we're going to finally unlock the true value of data, we're going to need to take a different approach.

We need to shift our mindset to one where *everyone* understands the value of data - from data professionals to product engineers to product managers. And because they understand the value, they're willing to put a bit more *discipline* into the creation of the data, at source, to prevent poor data quality impacting downstream applications.

This new approach is **data contracts**.

Improving data quality at source

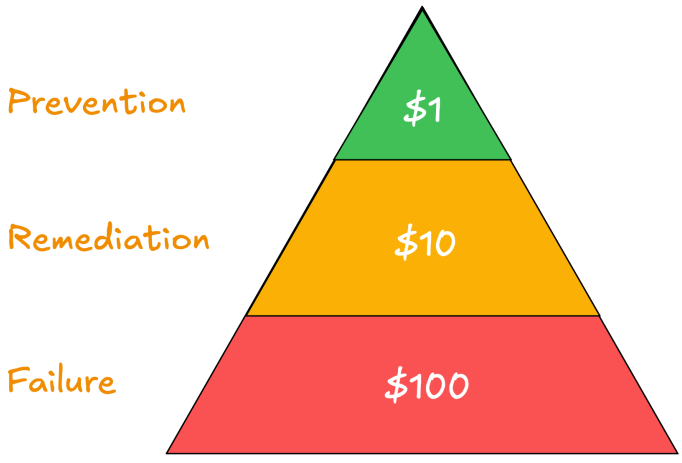
This new approach with data contracts is based on a simple truth: **Data quality can only be improved at the source**.

- If your data isn't timely enough at the source, you cannot make it more timely downstream
- If your data isn't accurate enough at the source, you cannot make it more accurate downstream
- If you don't have change management on your schemas at the source, you cannot provide stable schemas downstream

Not only is the source the only place to improve data quality, it's also the only place we can prevent data quality issues from occurring. And preventing data quality issues from occurring is far cheaper that resolving them once they have occurred.

This is illustrated by the 1:10:100 rule of data quality, which was developed by George Labovitz and Yu Sang Chang back in 1992 and states the following:

- The cost of preventing poor data quality at the source is $1 per record
- The cost of remediation after it is created is $10 per record
- The cost of failure (in other words, doing nothing) is $100 per record



The assertion that prevention is better than the cure is a universal rule.

Healthcare is an obvious example, where it's better for you and cheaper for the healthcare provider to prevent yourself getting sick by implementing good hygiene, taking vaccinations, and so on. It gets more expensive if you have to see a local doctor, and significantly more expensive if you end up needing hospital care.

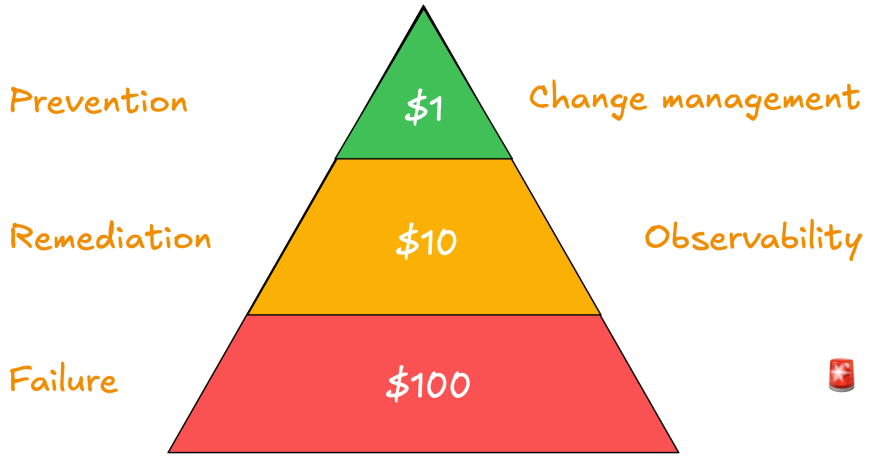
An example closer to the data domain is software engineering. Software engineers invest significantly to prevent bugs being deployed to production environments using techniques and tools such as:

- Code review
- Continuous integration
- Blue-green deployments

And many more, that collectively could be categorized as *change management*.

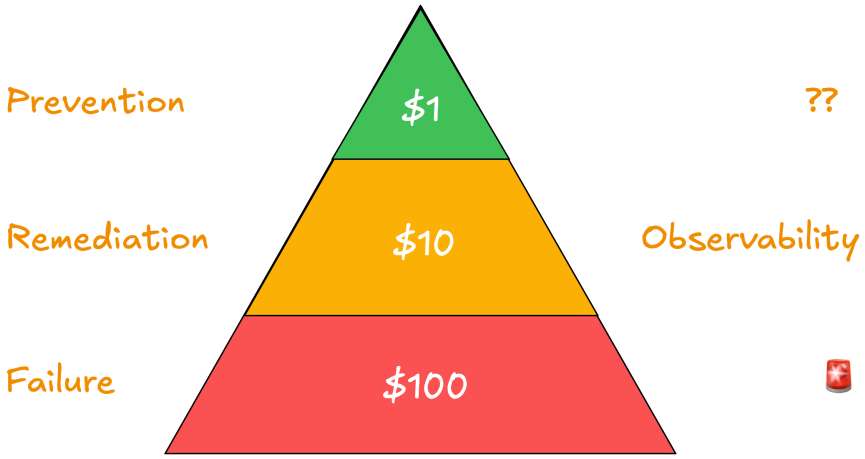
There's a cost to each of these, and it slows down the delivery of code to production, but the cost is accepted because the resulting quality is higher and the overall costs to the business are lower.

Of course, bugs still get deployed to production, so they still invest in observability, alerting, tracing, and so on. But it's known those remediation costs are significantly more than the investment in prevention, and so they continue to invest in change management.



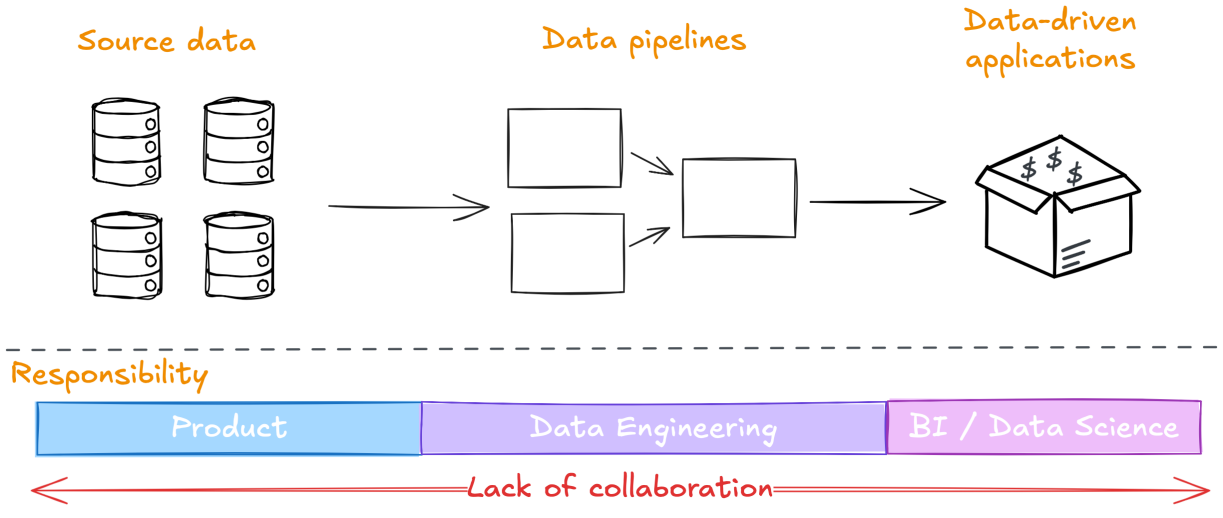
When it comes to data quality we have invested a lot in remediation with excellent solutions for observability, alerting, and root cause analysis now available from a variety of providers. That's good, because without that we'd *really* be struggling to do anything with our data, leading to failure, at a cost of $100 per record. But at $10 per record that's still at a significant cost.

We haven't invested much in preventing data quality issues *at source*, where the solutions would be cheapest and most effective. That's why today the costs and impact of poor data quality are so high.



So, how do we start preventing data quality issues, at source?

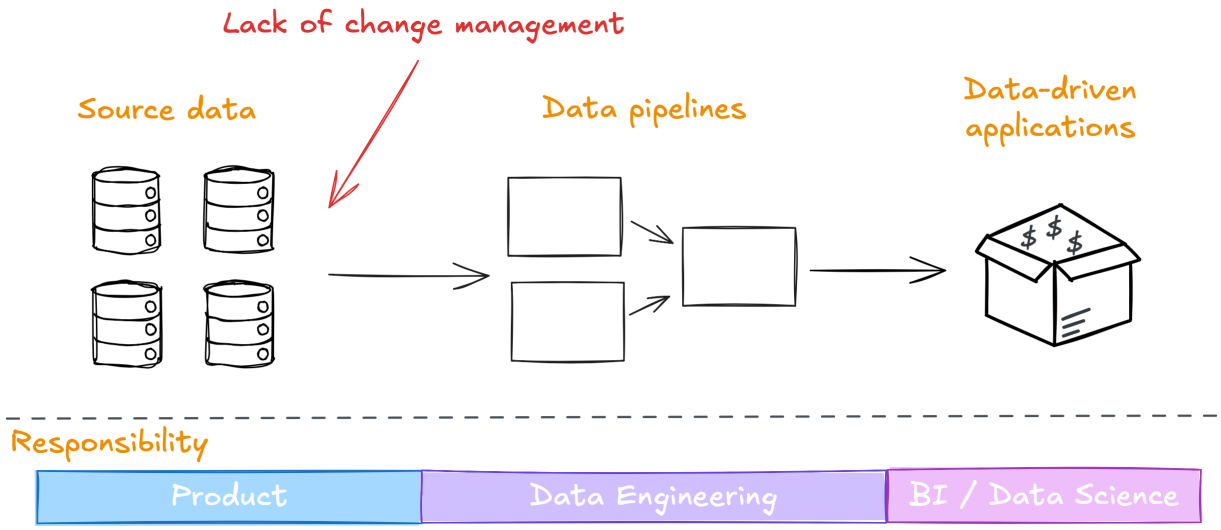
First, we need to solve the problem of responsibility, and in many organizations the people responsible for producing the data are the product engineers.



Now, they may not feel like that today! But they must be responsible, because only they have the ability to improve the quality of the data at the source. It's their code creating the data and it's their projects and changes that will impact the data over time.

Solving this problem requires us to greatly improve collaboration between those producing the data and those consuming it, and we'll look at exactly how data contracts solves that shortly.

The second problem we need to solve is the lack of change management at the source.



upstream database - same data, same schema, same changes.

We should be expecting that database to change as the product engineering teams add new features, improve performance, or perform any number of tasks needed to ensure their product is working effectively. And yet we are often caught out by this change.

We don't want to prevent these changes or slow down the product engineers, but we do need greater stability if we are going to build revenue generating data applications on this data, and do so with confidence.

Solving this requires a more explicit interface for the data. Data contracts provide that interface.

Let's look now at how data contracts solves these two problems.

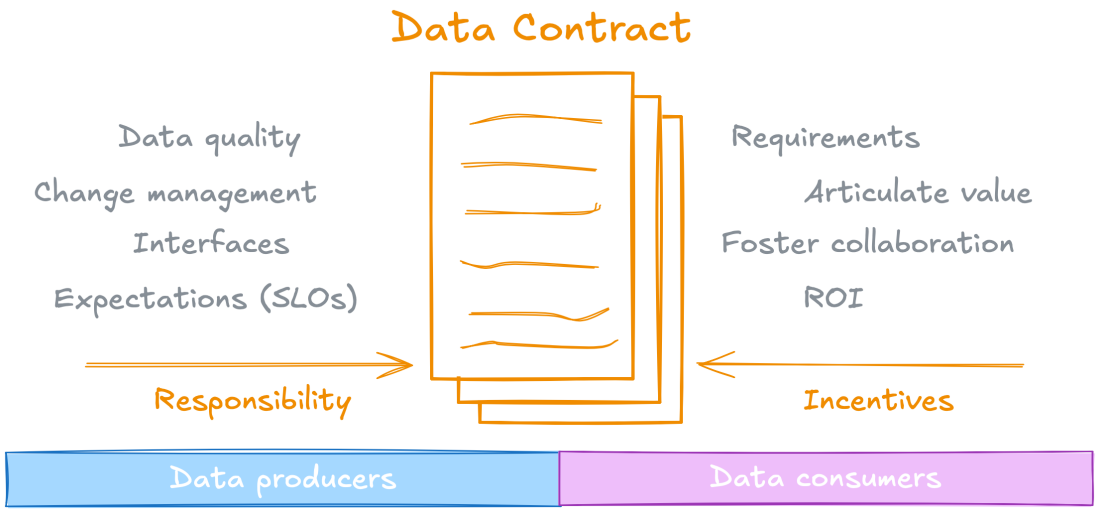
Encouraging collaboration

As mentioned earlier, those creating value from the application of data are often far away from those producing the data. If those producing the data don't see the value of the data, then why would they be incentivized to create better quality data or take on more responsibility for the stability of that data?

If data consumers are going to ask their product teams to take on more responsibility for the quality and dependability of data and incentivize them to do so, they need to be good at articulating the value of the data, and the positive outcomes they are generating for the organization through the application of that data.

This is the *why* they are doing it. Why is it worth the investment in improving data quality? What are the positive outcomes we are generating for the business? Why are we investing in data instead of something else?

With this understanding, the data producer is more incentivized to provide a quality data that meets their requirements and gains a sense of ownership of, and responsibility for, the business value they are helping to create.



The data contract then becomes the place these agreements are recorded and codified, including some or all of the following:

- Ownership and responsibilities
- The structure/schema of the data
- The valid/invalid data values (data quality checks)
- How performant and dependable the data is (via SLOs)

Now this has been codified in a data contract we can apply change management to it, as discussed next.

Explicit interface

Interfaces are powerful. That's why we see them everywhere in software engineering. In fact, I'd say they are essential when you need to depend on something provided by someone else.

For example, when you use a software library in your project, you're using its public interface. It's well documented, versioned, and easy to use. You know that you can build on it with confidence and it's going to be pretty stable. If/when it does need to change there will be a new major version of the library released with a migration path for you to follow, when you are ready to do so.

Of course, none of these guarantees are provided if you're using the libraries private interface, for example methods that have not been marked as public.

APIs (application programming *interfaces*) are another example, and are used internally between services and externally between organizations. People build entire businesses upon APIs from organizations like Stripe, Slack and Github and can do so with confidence.

Data typically isn't provided through an interface. It's generally an extraction of an upstream database to the data warehouse or data lake which we then build pipelines on. That database is the upstream services *private* interface, so of course we don't get any guarantees on the dependability or the quality of the data when we build on it, nor any change management.

With data contracts we provide data through an explicit interface that:

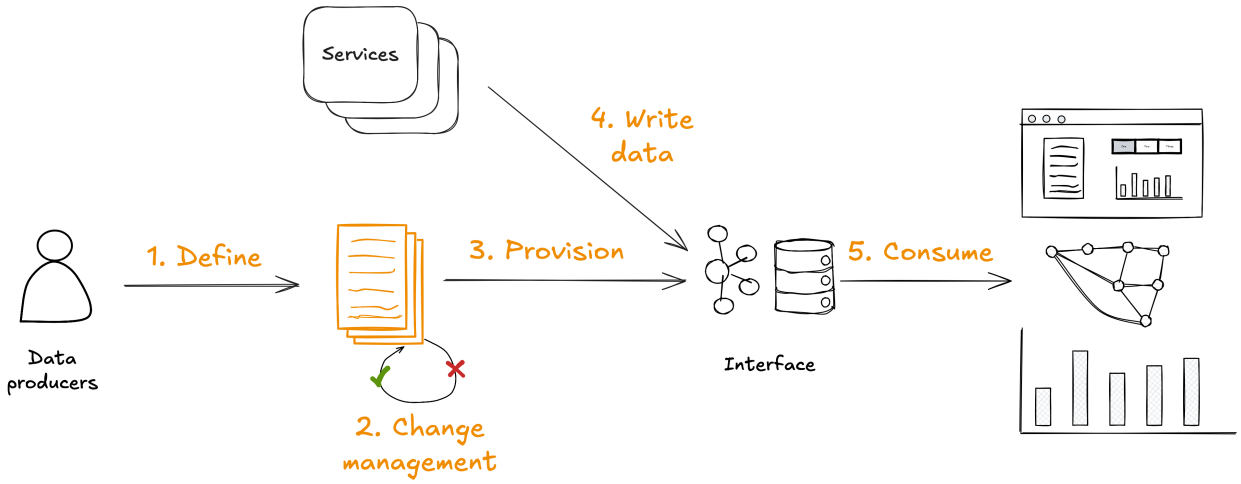
- Is decoupled from the upstream database, and therefore more stable even as it changes
- Presents the data in a form that is easier for consumers to use
- Has effective change management applied

That interface could be a table in a data warehouse, an area in a distributed filesystem, a streaming pipeline, or however your users need to consume data.

The upstream services are then modified to write the data to that interface, and consumers consume only from that interface.

In practice this is achieved through 5 steps:

1. The data producers define the data contract, as agreed with the data consumers
2. Change management is applied to that data contract, preventing breaking changes
3. The data contract provisions the interface, typically through an infrastructure as code (IaC) tool
4. Services write the data directly the the interface, which doesn't need to look anything like their internal database structure
5. Users consume from the interface, with the confidence they can invest in building great applications on top of it



Of course, there is a higher upfront cost to building these interfaces and modifying your upstream services to adopt this new approach. But this is an investment in preventing data incidents and will be paid back by reducing the costs associated with incidents, the costs of complex ETL, and the opportunity cost of being unable to deliver the data applications your business needs to achieve its strategic goals.

A change in approach

As the importance of data and the applications we build upon them increases we need to apply more discipline to the generation of our data at the source.

We're asking data producers to do more, and we're doing so because we can prove the it reduces overall costs for the business and leads to better outcomes.

Change isn't easy. It's going to take time. More than anything else, your success will be determined by your communication. That includes:

- Using different formats for different results (newsletters, workshops, tech talks, meetings, etc)
- Building relationships with people in different teams and roles
- Using the right language for the audience

You could have the best data contract implementation, but without communication your adoption of data contracts, or any transformation project, will not be a success.

But I've seen first-hand that this is achievable at organizations in the US, Europe, and Asia, from startups to enterprises, and in industries as varied as tech, finance and construction. All of their data is different, but the problems are the same, and data contracts are part of the solution to those problems.

So trust me, this will be worth it.

Because when it comes to data quality, **prevention is better than the cure**.

What's next

In this Data Contracts 101 eBook we only scratched the surface of the power of data contracts. We wasn't able to cover in detail some of the other benefits, including:

- Embedding and automating data governance
- Providing a self-served, contract-driven data platform
- Using data contracts to implement a data mesh architecture

I cover all that, and more, in my book *Driving Data Quality with Data Contracts*, available at https://data-contracts.com.

I also provide a free newsletter providing tips on implementing data contracts and related subjects. You can sign up at https://andrew-jones.com/newsletter.

I also run workshops on implementing data contracts. Find out more at https://andrew-jones.com/workshops.

I love talking about data contracts! So please feel free to book a no obligation brainstorming call at https://andrew-jones.com/chat or if you prefer email me at andrew@andrew-jones.com.

Thank you for taking the time to read this white paper.

About the author

Andrew Jones is a Principal Engineer, instructor, and author. He helps data leaders transform their organization to one where data *drives revenue* - is dependable, governed, and valuable - and is guaranteed with **data contracts**.

In 2021 he created data contracts, an architectural pattern that brings data teams closer to the business, drives team autonomy, and embeds quality and governance as part of the data platform.

In 2023 he wrote the definite book on data contracts, which currently has an average rating of 4.7 stars on Amazon.

He is based outside London, UK, and is a regular writer and public speaker.

andrew@andrew-jones.com https://andrew-jones.com