



Data Engineering Rises Again

Escaping The Modern Data Trap With
Composable, Embeddable Software

Nick Schrock
Founder, Dagster Labs

What is the modern data stack?

What is the modern data stack?



The modern data stack (MDS) is a suite of tools used for data integration. These tools include, in order of how the data flows:

- a fully managed **ELT data pipeline**
- a cloud-based columnar warehouse or data lake as a destination
- a data transformation tool
- a business intelligence or data visualization platform.

What is the modern data stack?



Ultimately, the modern data stack lowers the technical barrier to entry for data integration. The components of the modern data stack are built with analysts and business users in mind, meaning that users of all backgrounds can not only easily use these tools, but also administer them without in-depth technical knowledge.

The Modern Data Stack has reshaped the data ecosystem with extremely positive benefits.

The Modern Data Stack has reshaped the data ecosystem with extremely positive benefits.

However, one goal was to **eliminate data engineering.**

The reality

The reality

Data engineers are
more important
than ever

The reality

Data engineers are
more important
than ever

They are poorly
served by the current
modern data stack

Where the MDS succeeded

Where the MDS succeeded

 Easy

Spin-up to get started/prototype

Where the MDS succeeded

 **Easy**

Spin-up to get started/prototype

 **A big tent**

Can get lots of stakeholders involved

Where the MDS succeeded

✓ Easy

Spin-up to get started/prototype

✓ Reduced incidental complexity

- dbt-core, not custom templated SQL
- No one wants to write data movement code

✓ A big tent

Can get lots of stakeholders involved

Where the MDS succeeded

✓ Easy

Spin-up to get started/prototype

✓ Reduced incidental complexity

- dbt-core, not custom templated SQL
- No one wants to write data movement code

✓ A big tent

Can get lots of stakeholders involved

✓ Cloud-native

Reduced fixed cost and better DX

Where the MDS falls apart

Where the MDS falls apart

- ⊗ Presumes homogenous storage and compute

Where the MDS falls apart

⊗ Presumes homogenous storage and compute

⊗ Too many operational silos

Where the MDS falls apart

⊗ Presumes homogenous storage and compute

⊗ Too expensive

⊗ Too many operational silos

Where the MDS falls apart

⊗ Presumes homogenous storage and compute

⊗ Too expensive

⊗ Too many operational silos

⊗ Quickly descends into chaos

Where the MDS falls apart

⊗ Presumes homogenous storage and compute

⊗ Too expensive

⊗ Too many operational silos

⊗ Quickly descends into chaos

⊗ Poor automation, flexibility, and composability

Presumes homogenous storage
and compute

Presumes homogenous storage and compute



Matthew Mullins

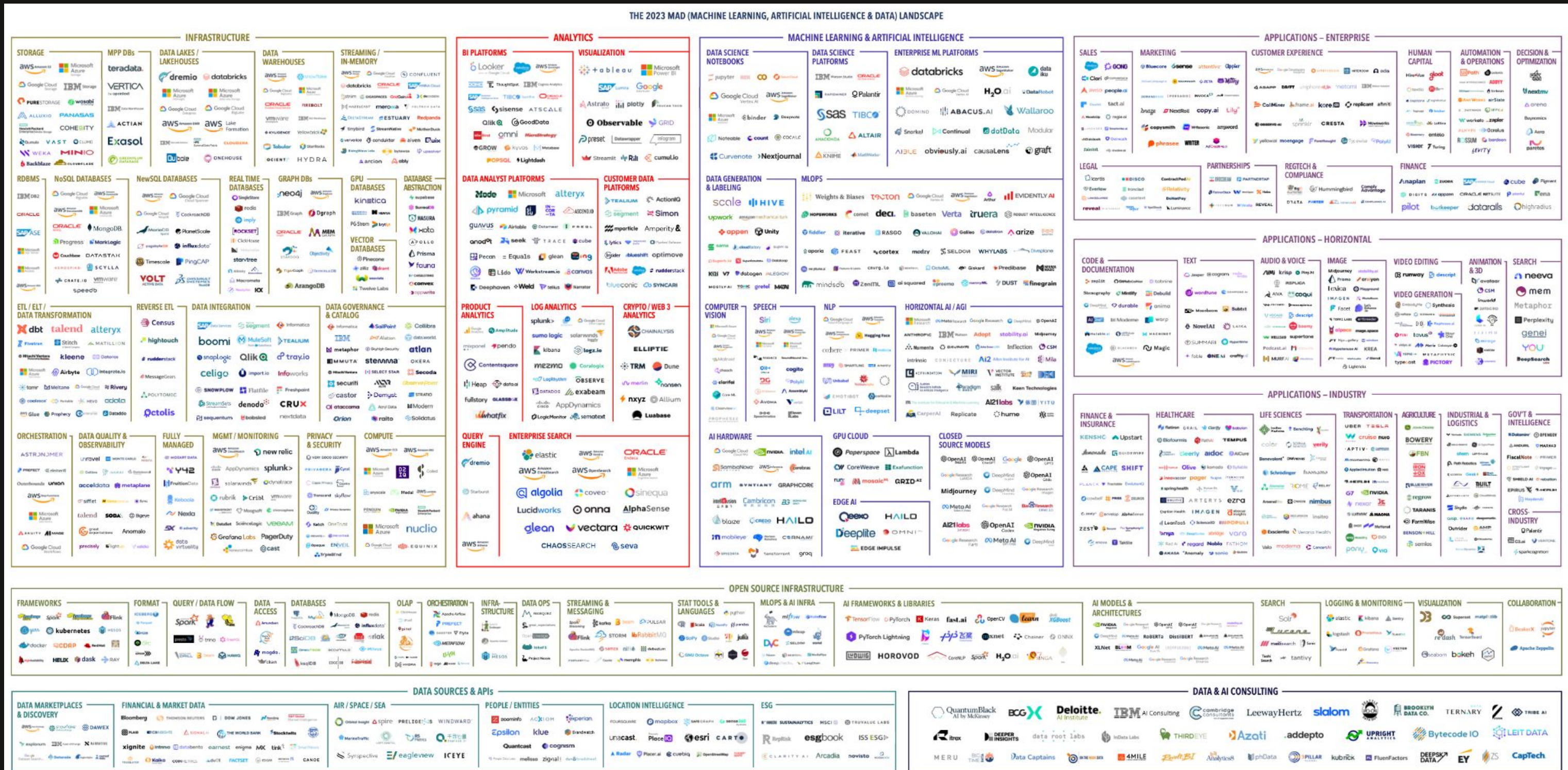
@mullinsms



The lie of every MDS architecture diagram is that there is a single warehouse at the center of the business. I've almost never seen such an architecture.

Too many disconnected tools

Too many disconnected tools



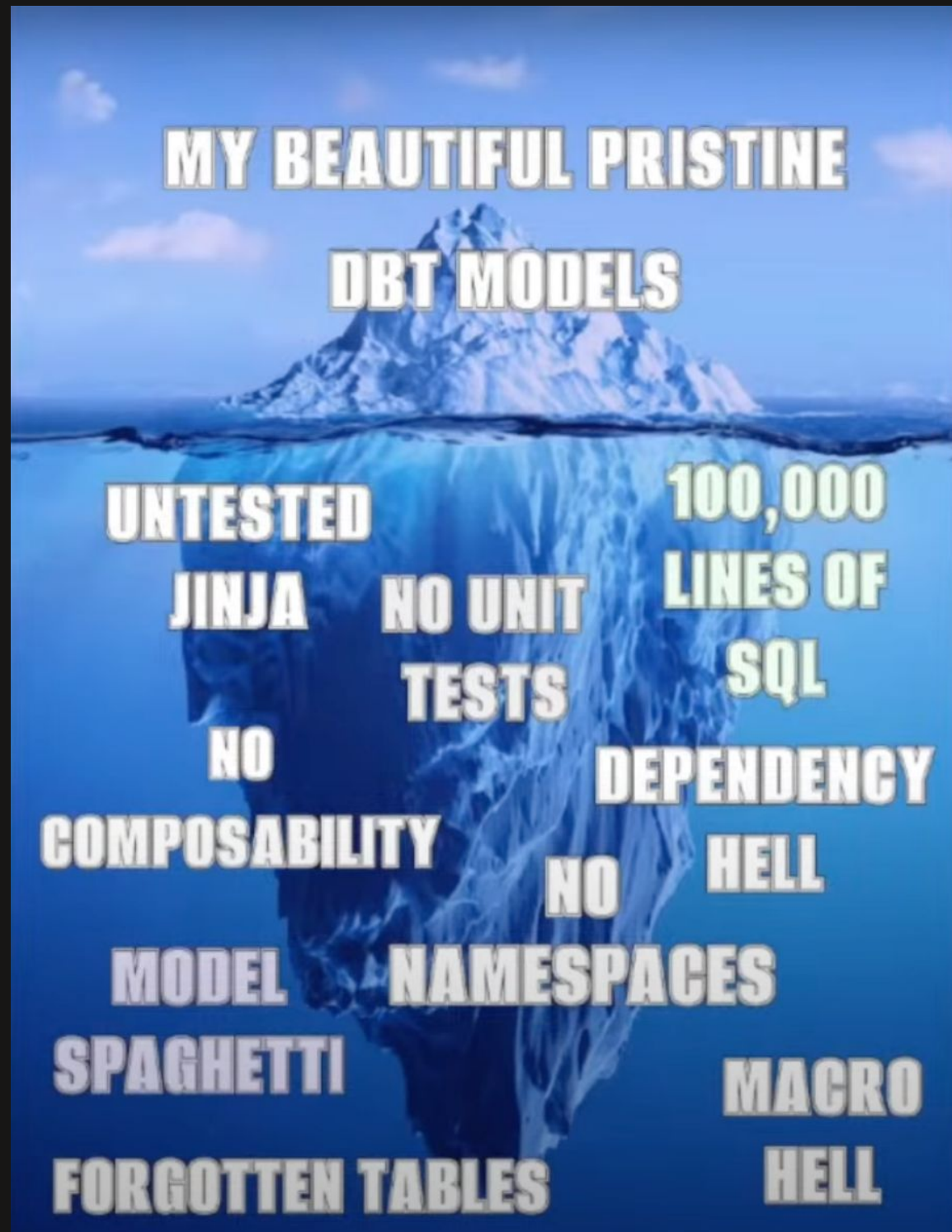
Too expensive

Too expensive



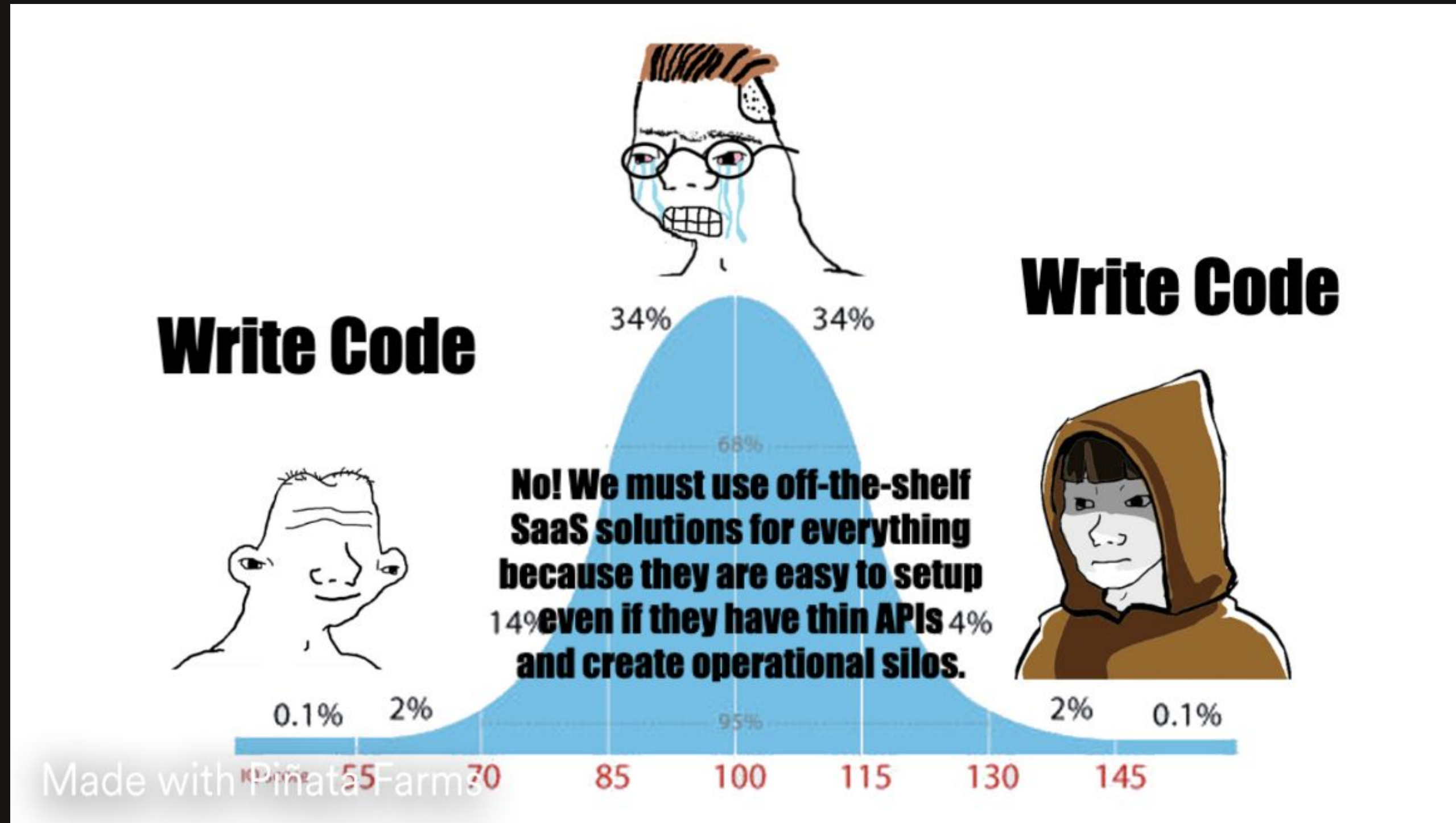
Descends into chaos

Descends into chaos

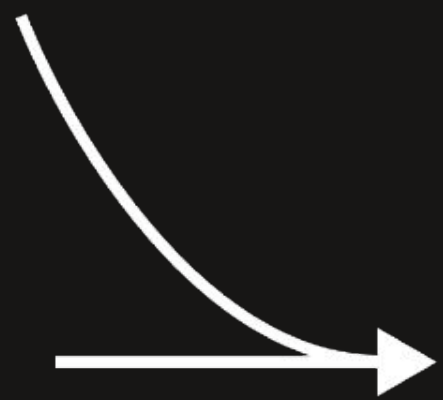


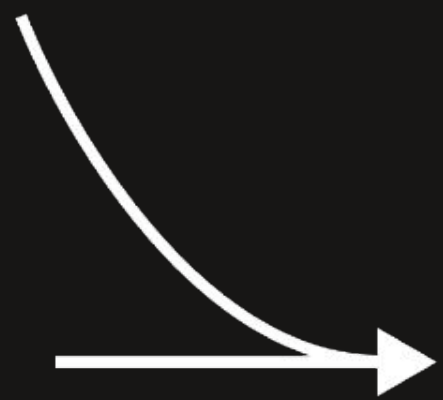
Poor automation, flexibility, composability

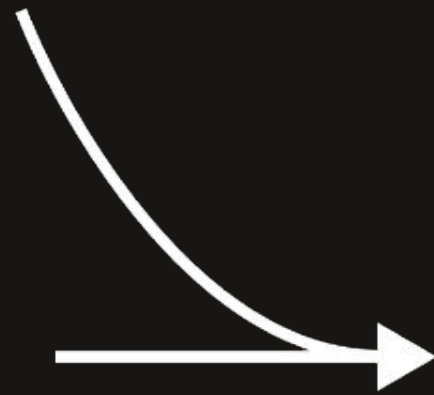
Poor automation, flexibility, composability

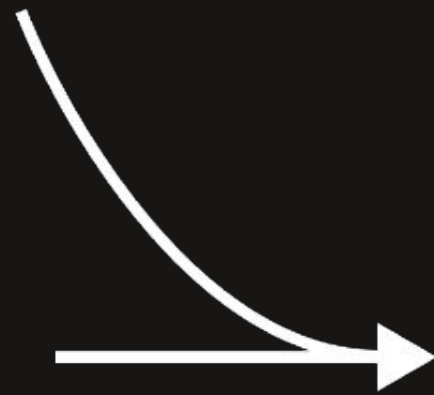


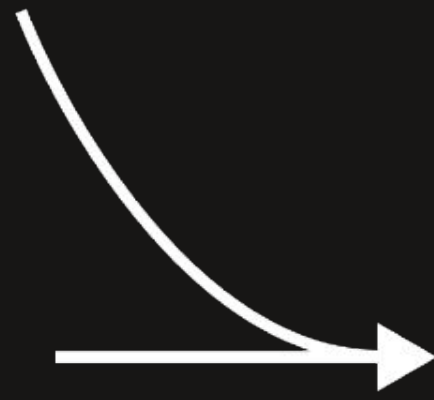


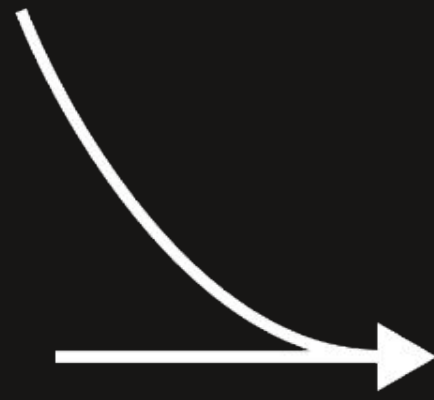


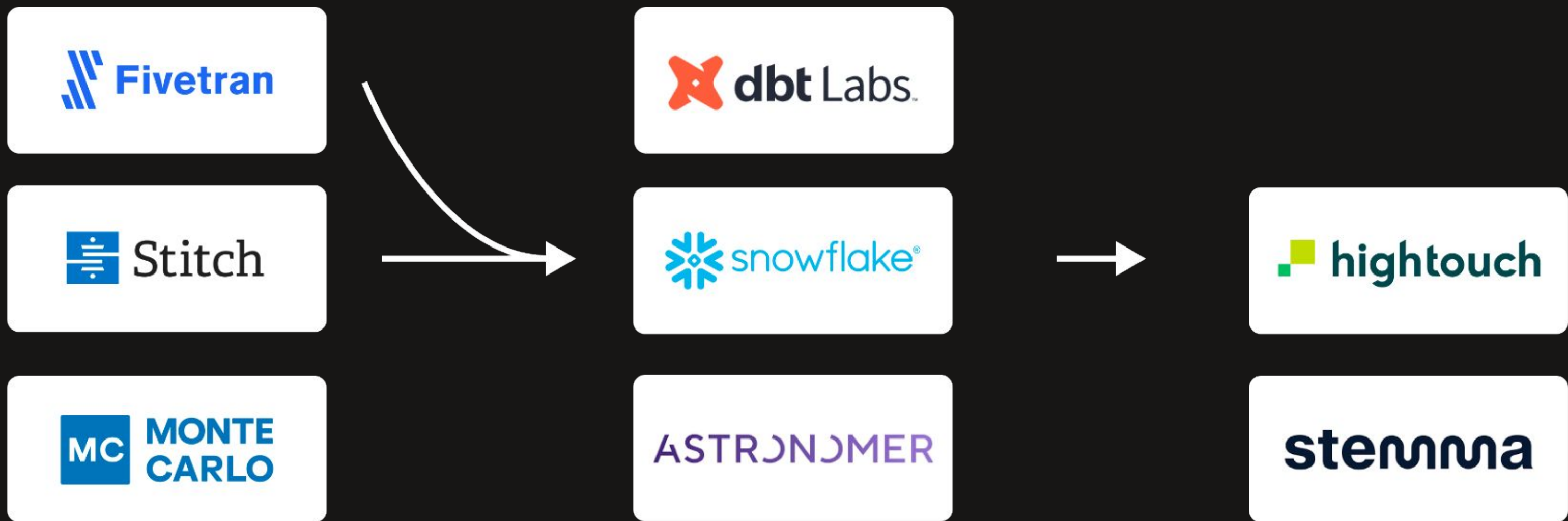
 **Fivetran** **Stitch** **dbt Labs.** **snowflake®**



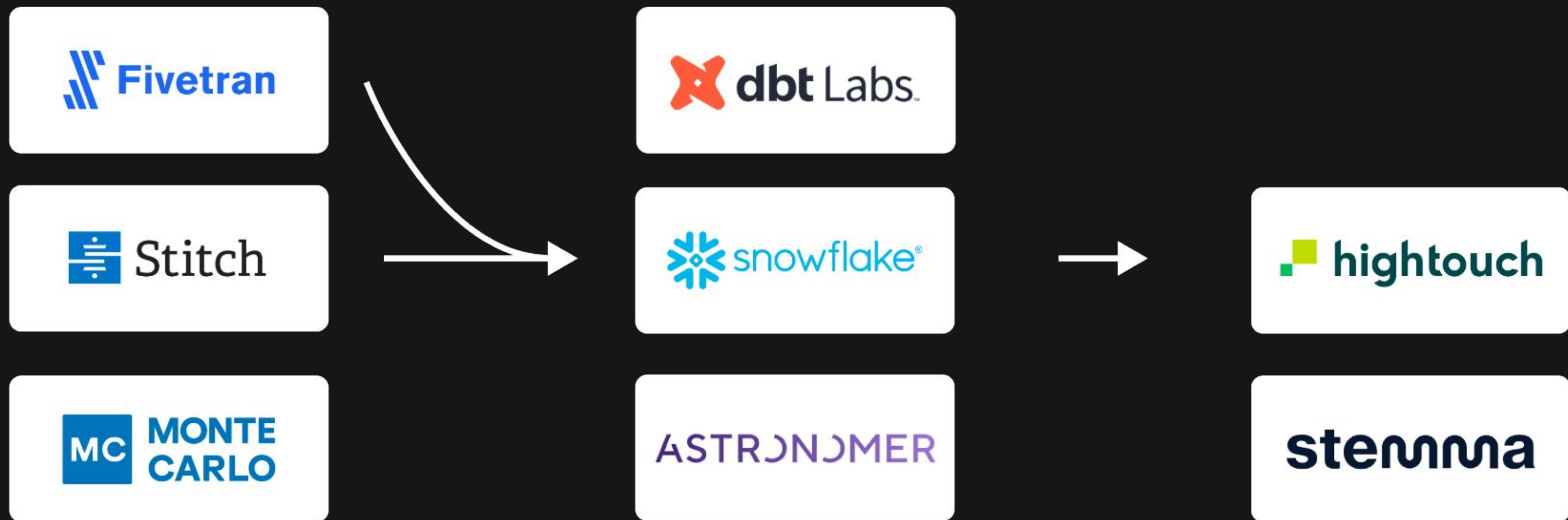








All this to build a single quality end-to-end data pipeline with searchable metadata



All this to build a single quality end-to-end data pipeline with searchable metadata

- Seven vendors to negotiate
- Four schedulers to babysit
- Six UIs to operate

Standalone tools or embedded
libraries?

Standalone tools or embedded libraries?



Standalone tools or embedded libraries?

 PostgreSQL

 snowflake®

 SQLite

 DuckDB

vs.

 Fivetran

 MC MONTE CARLO

 Sling

 great expectations

Standalone tools or embedded libraries?

 PostgreSQL

 snowflake®

 SQLite

 DuckDB

vs.

 Fivetran

 MC MONTE CARLO

 Sling

 great expectations

Depends on context!

Standalone services and tools can

Standalone services and tools can

Overserve engineers

Could be just embedded libraries

Standalone services and tools can

Overserve engineers

Could be just embedded libraries

Underserve engineers

Thin APIs, inflexible, hard to automate

Standalone services and tools can

Overserve engineers

Could be just embedded libraries

Be operationally fragile

Ship with thin, underpowered orchestrators

Underserve engineers

Thin APIs, inflexible, hard to automate

Standalone services and tools can

Overserve engineers

Could be just embedded libraries

Be operationally fragile

Ship with thin, underpowered orchestrators

Underserve engineers

Thin APIs, inflexible, hard to automate

Have unpredictable costs

Many vendors, many pricing models, poor visibility

Are standalone services
the only option?

**Adopting off-the-shelf SaaS
tools without a unified control
plane**

It's a trap





**Adopting off-the-shelf SaaS
tools without a unified control
plane**

It's a trap

**“Easy” but disconnected tools often end up
quickly creating more complexity**

Escaping the Trap

A Flexible, Composable Single Pane of Glass

Escaping the Trap

A Flexible, Composable Single Pane of Glass



Builds on the good parts of the
MDS

Escaping the Trap

A Flexible, Composable Single Pane of Glass



Builds on the good parts of the
MDS



Embraces engineering best
practices

Escaping the Trap

A Flexible, Composable Single Pane of Glass



Builds on the good parts of the MDS



Embraces engineering best practices




A composable toolkit for data engineers

Escaping the Trap

A Flexible, Composable Single Pane of Glass

 Builds on the good parts of the MDS

 Embraces engineering best practices

 A composable toolkit for data engineers


 Single pane of glass for the entire data platform

Automation, observability, lineage, data quality, and consumption management


Escaping the Trap

A Flexible, Composable Single Pane of Glass

 Builds on the good parts of the MDS

 Embraces engineering best practices

 A composable toolkit for data engineers

 Single pane of glass for the entire data platform
Automation, observability, lineage, data quality, and consumption management

 Incremental adoption across every stakeholder team



dagster



Thank you