







# Rethinking the DAG

Escaping the workflow ↔ data platform  
impedance mismatch

Pete Hunt  
CEO - Dagster Labs

# Agenda

- About me
- The mess we're in
  - Developer velocity
  - Stack complexity
  - Data decentralization
- The impedance mismatch
- Rethinking the DAG and its implications



# Who am I?



## CEO at Dagster Labs

- Dagster Core (OSS)
- Dagster+ (coming April 17)



Ran data teams at Twitter and co-founded a streaming data company for detecting online abuse



Founding member of the React project at Facebook

**The mess we're in**

Imagine yourself as a new Head of  
Data at a Series B company building  
a Spotify competitor





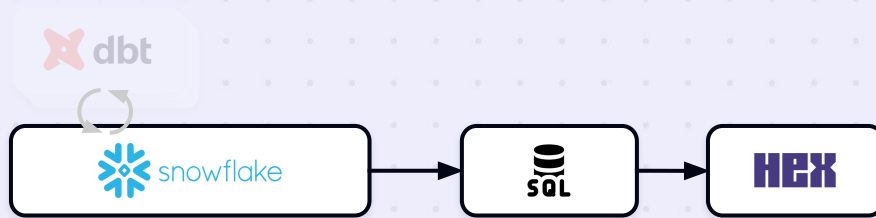


Lydia / Analytics Engineer



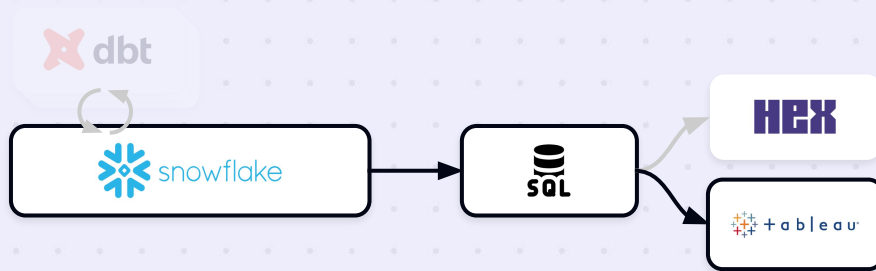


Olivia / CEO



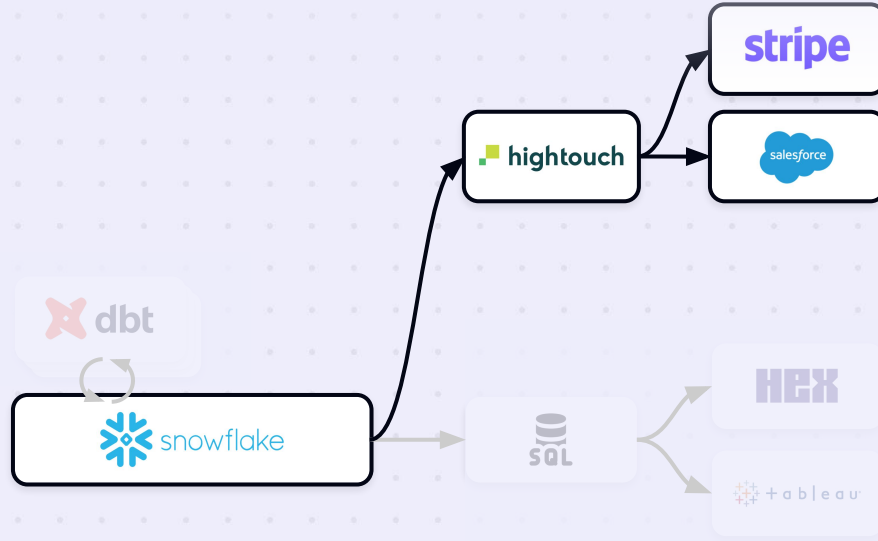


Ed / Head of Marketing





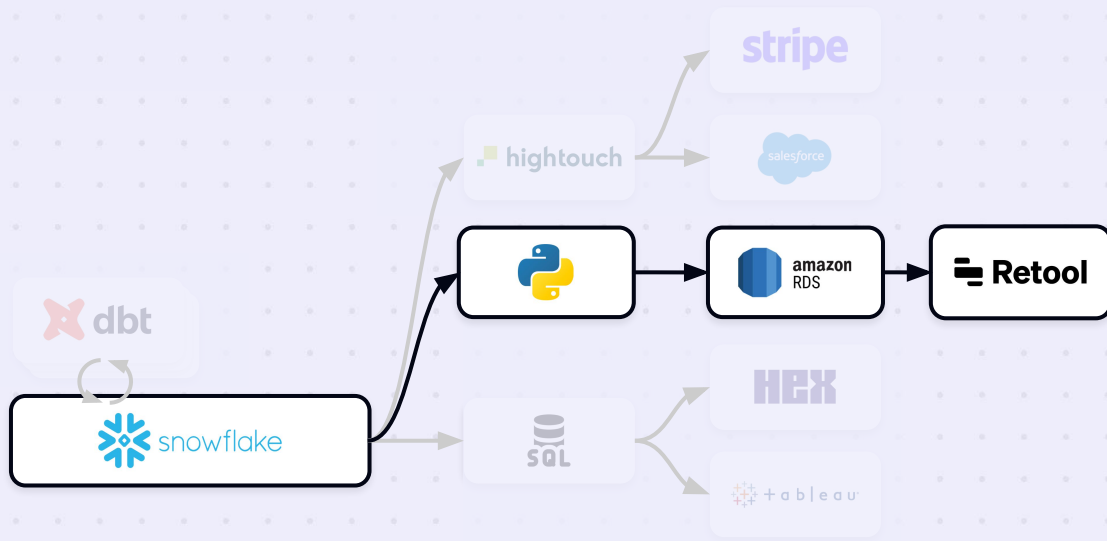
## Wade / Head of Sales





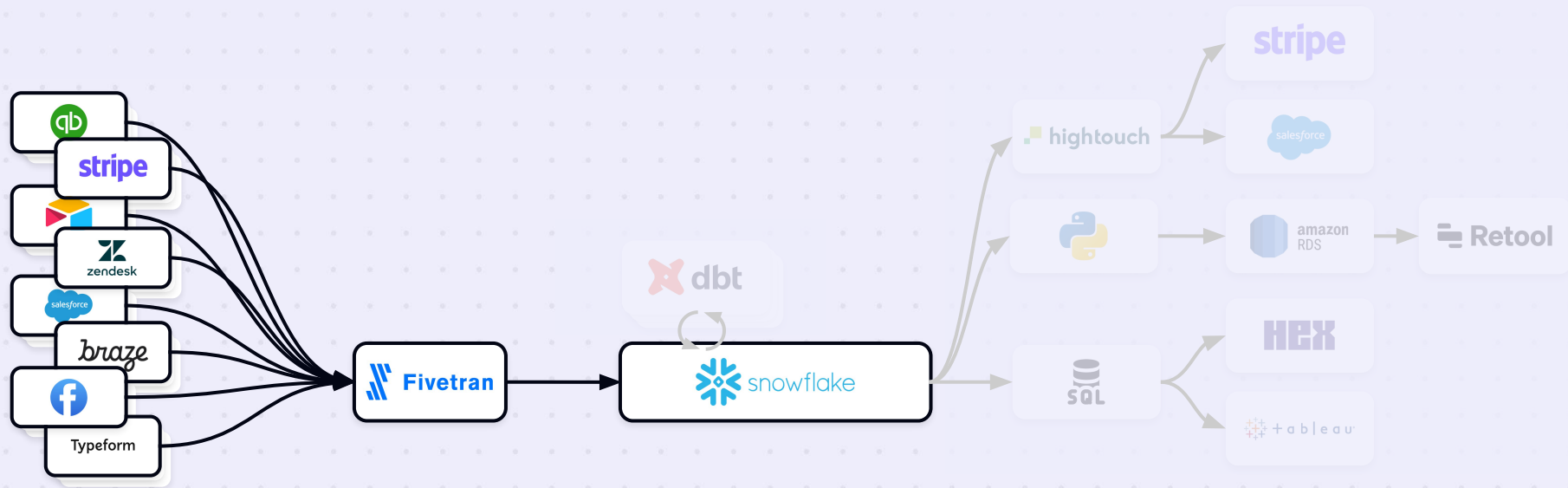


## Joel / Customer Success Manager



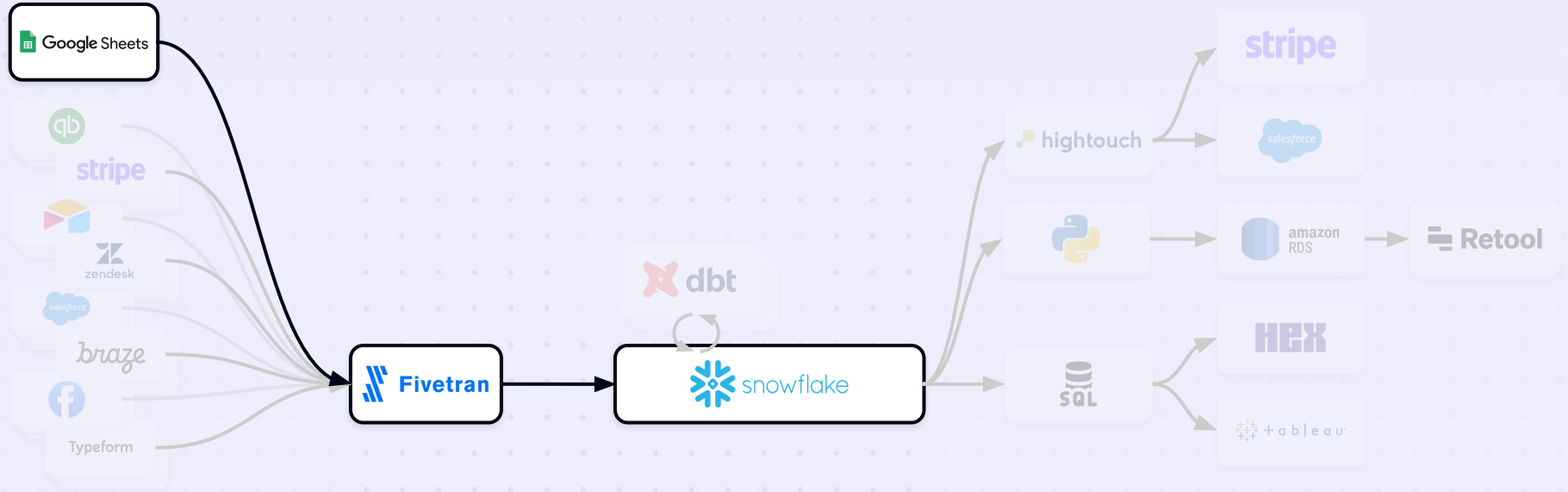


## Joel / Data Contractor at ACME Analytics



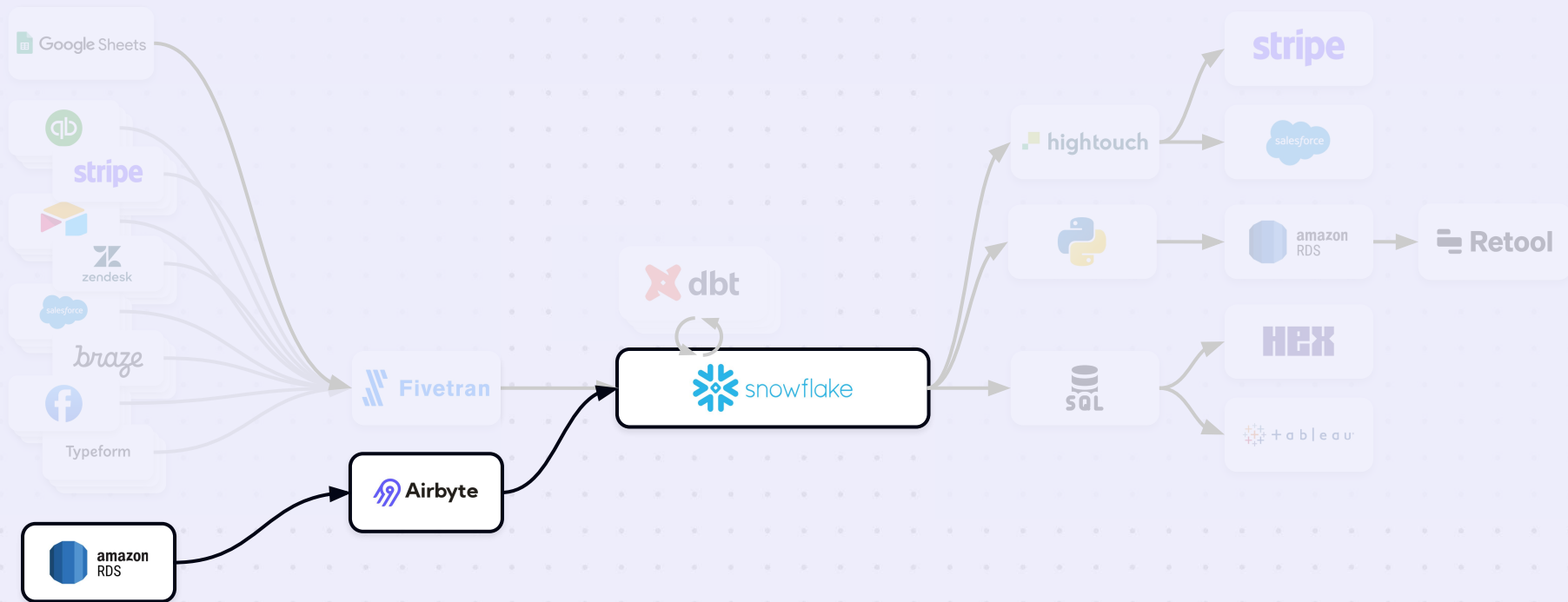


## Grace / Product Manager



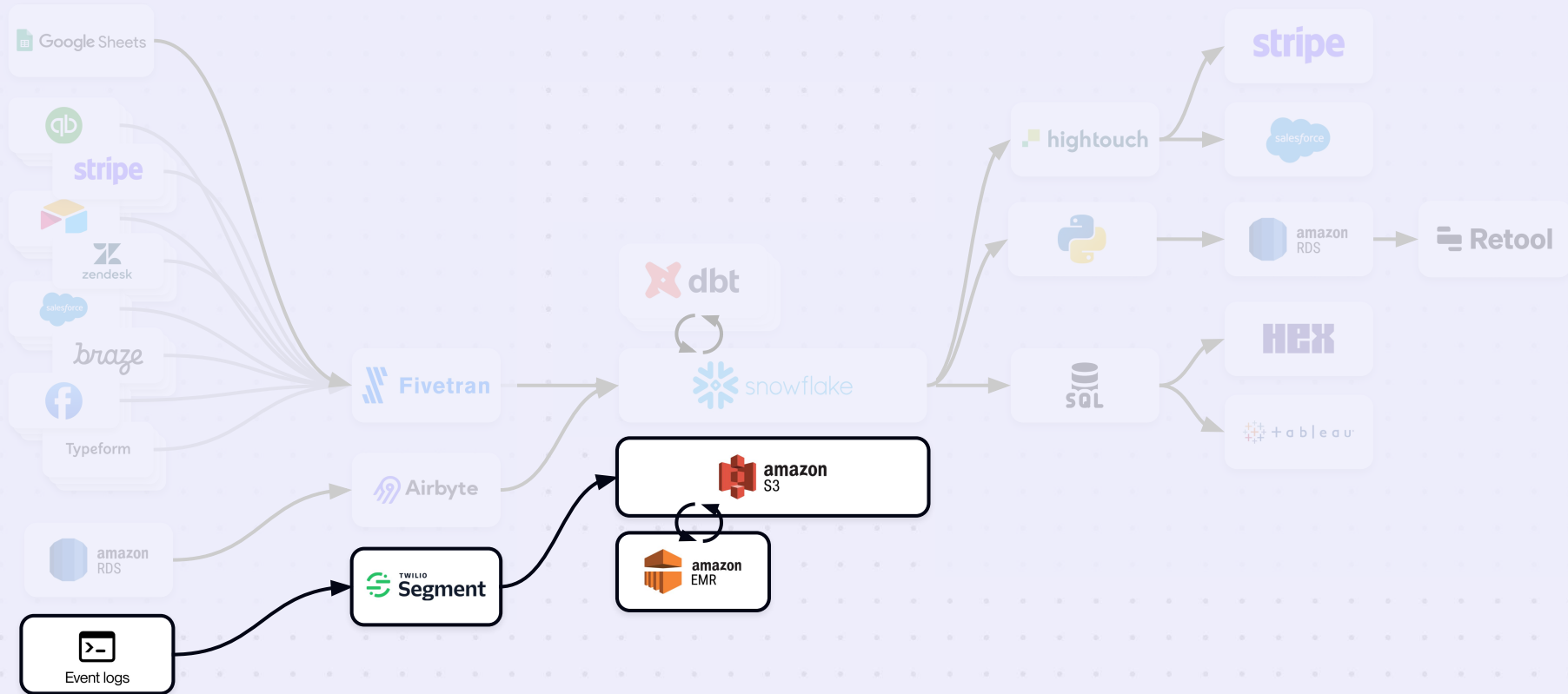


# Peyton / Data Engineer



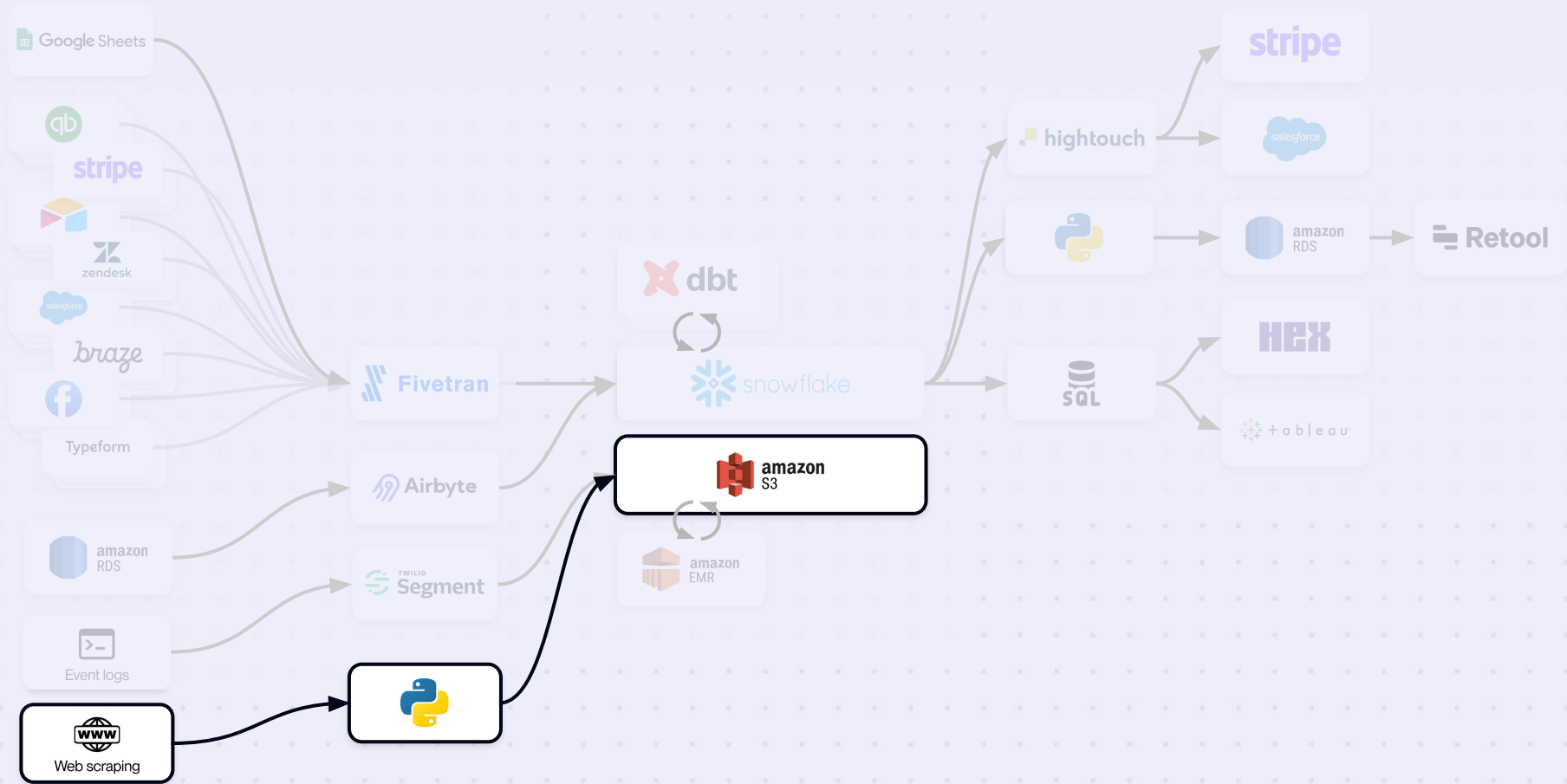


## Carlie / Product Engineer



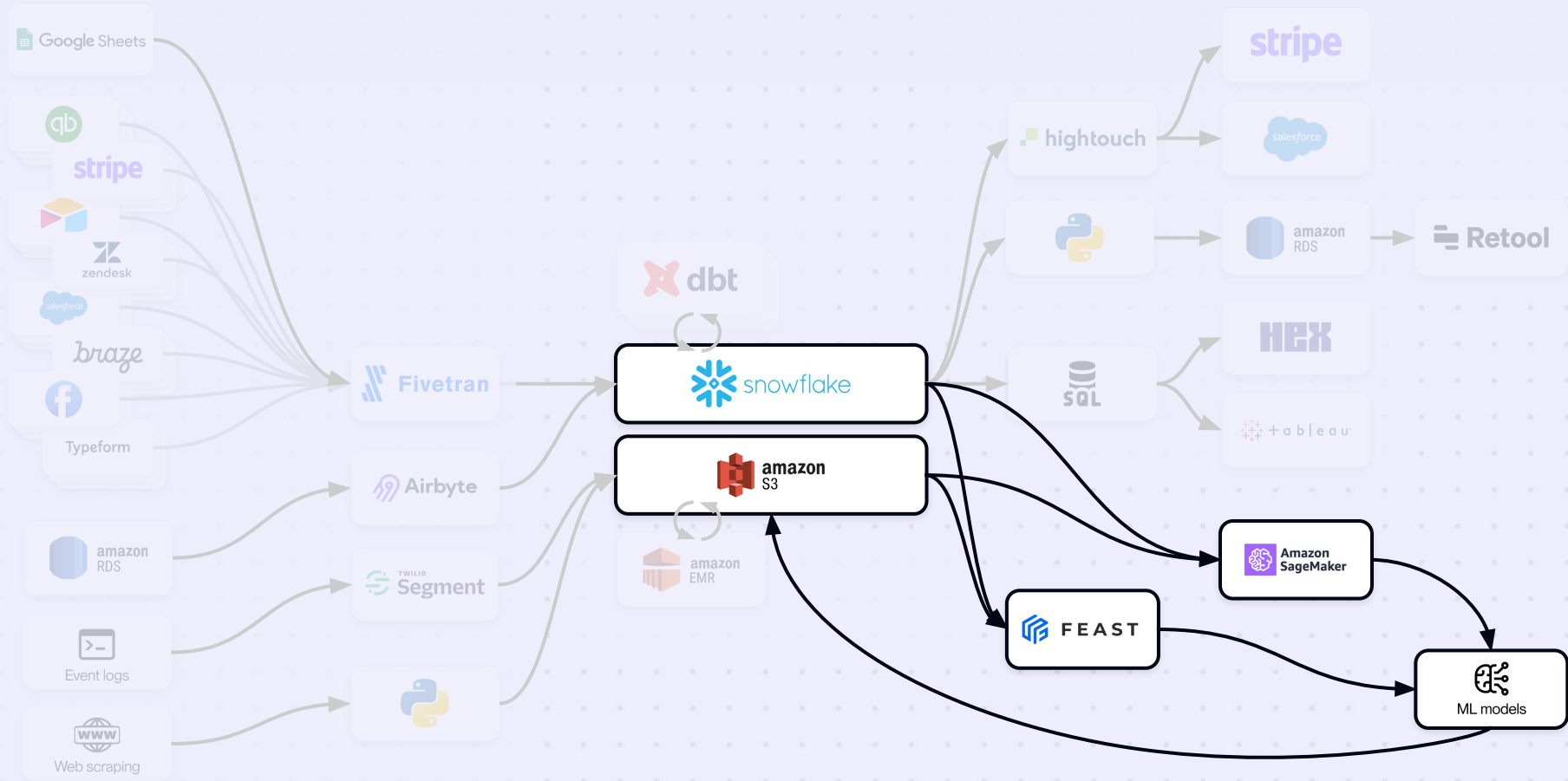


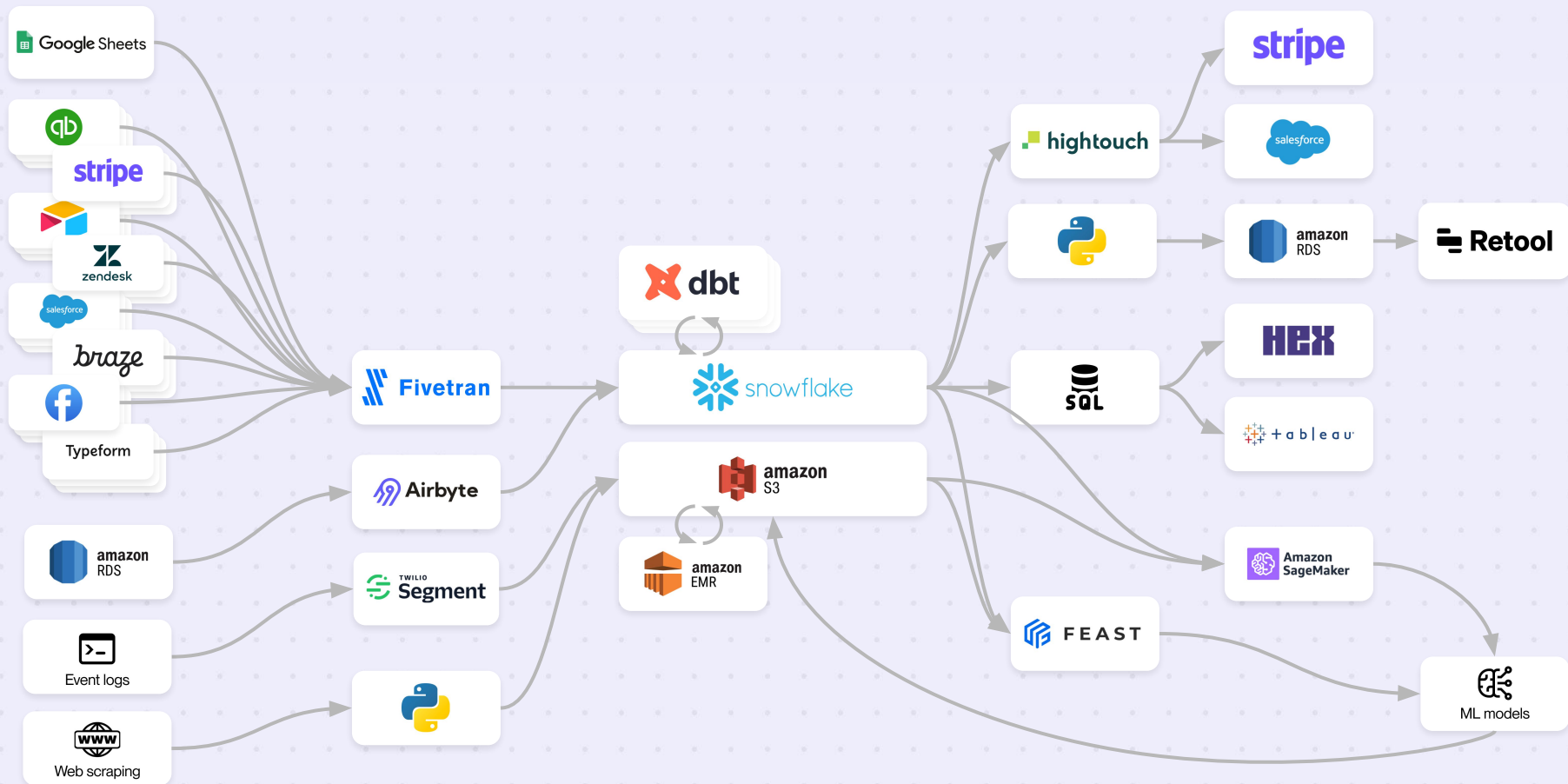
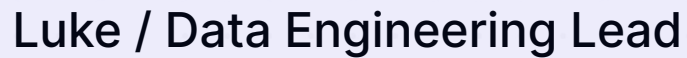
## Oscar / ML Engineer



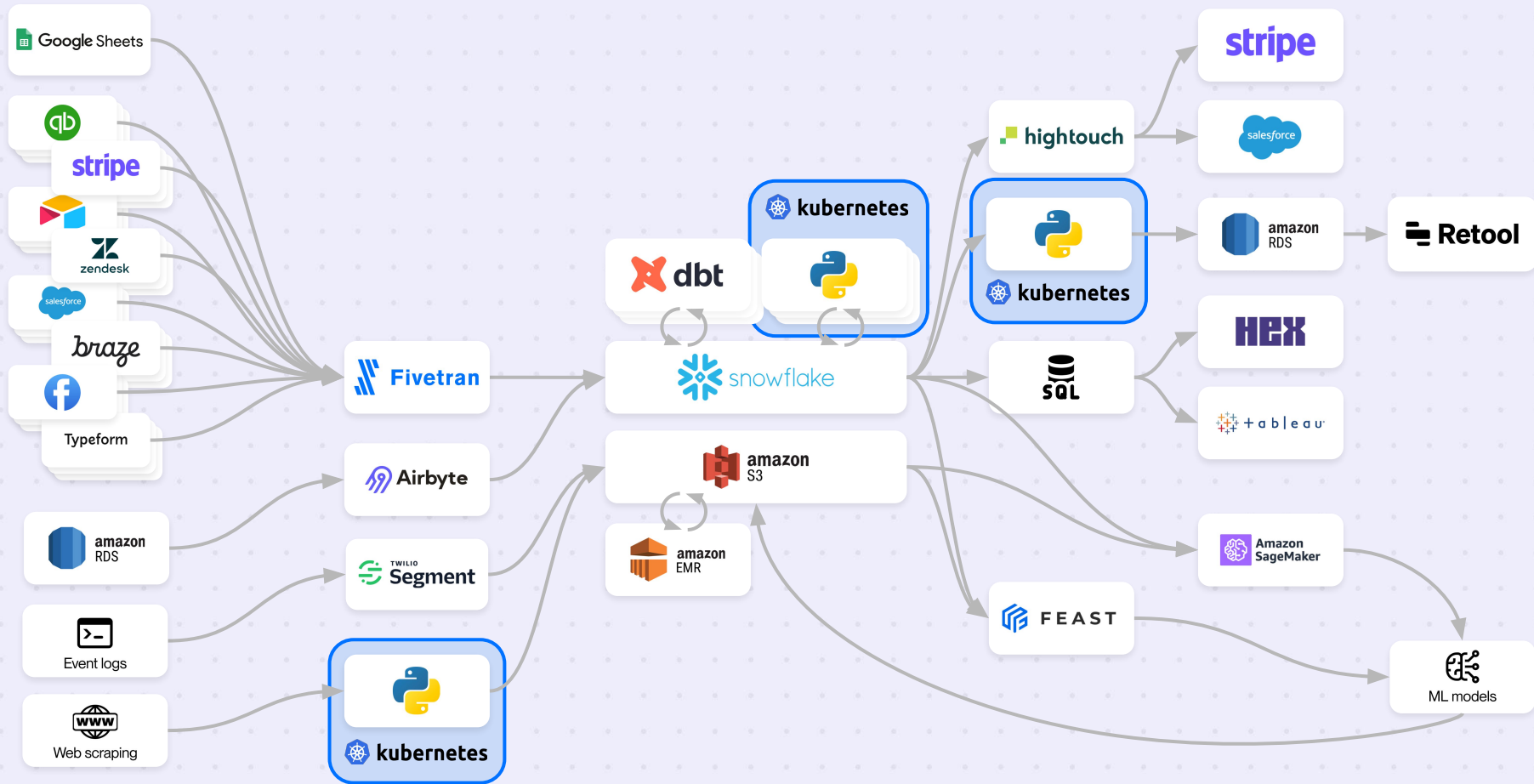
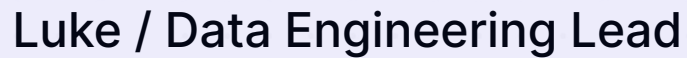


# Hector / Data Scientist



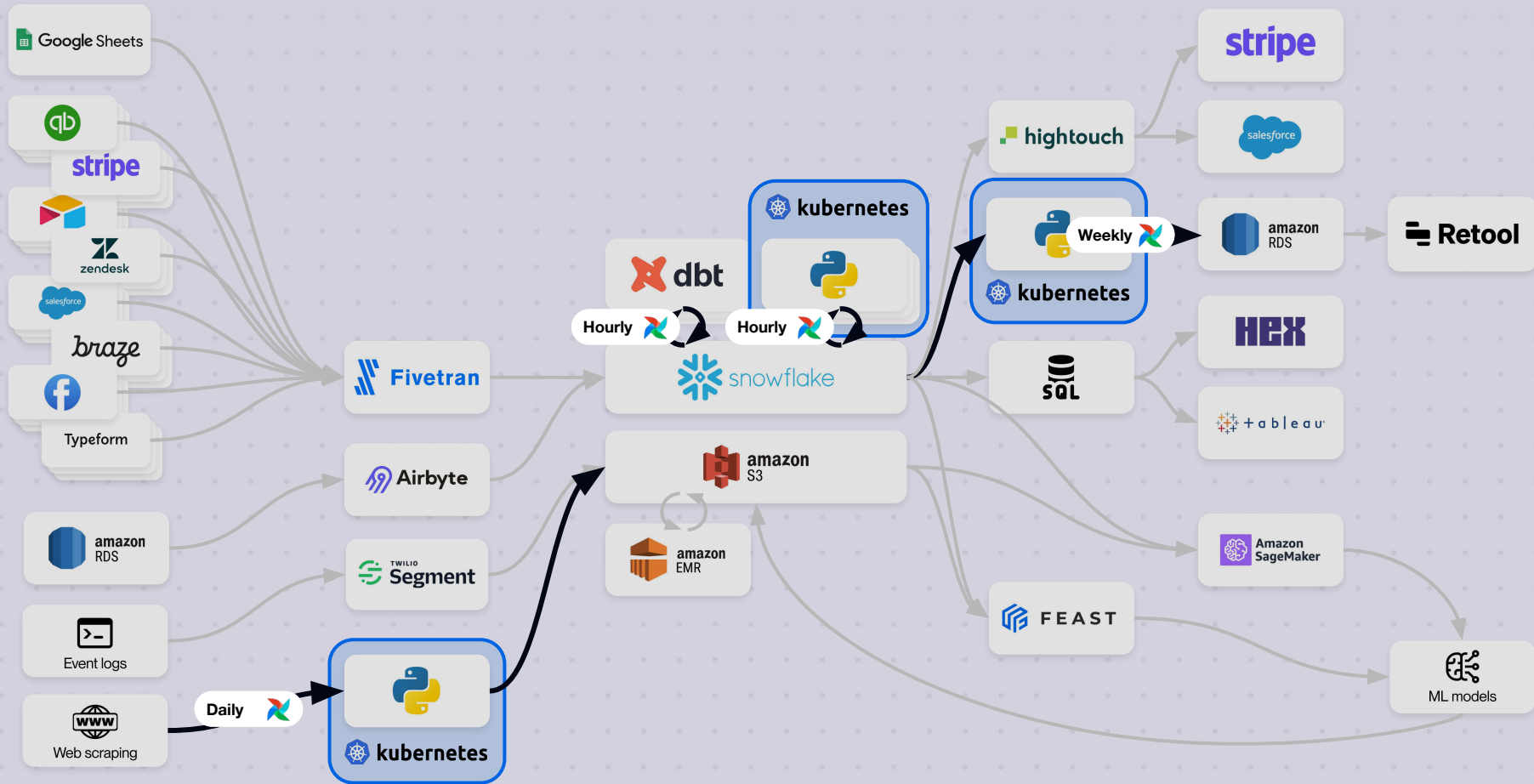






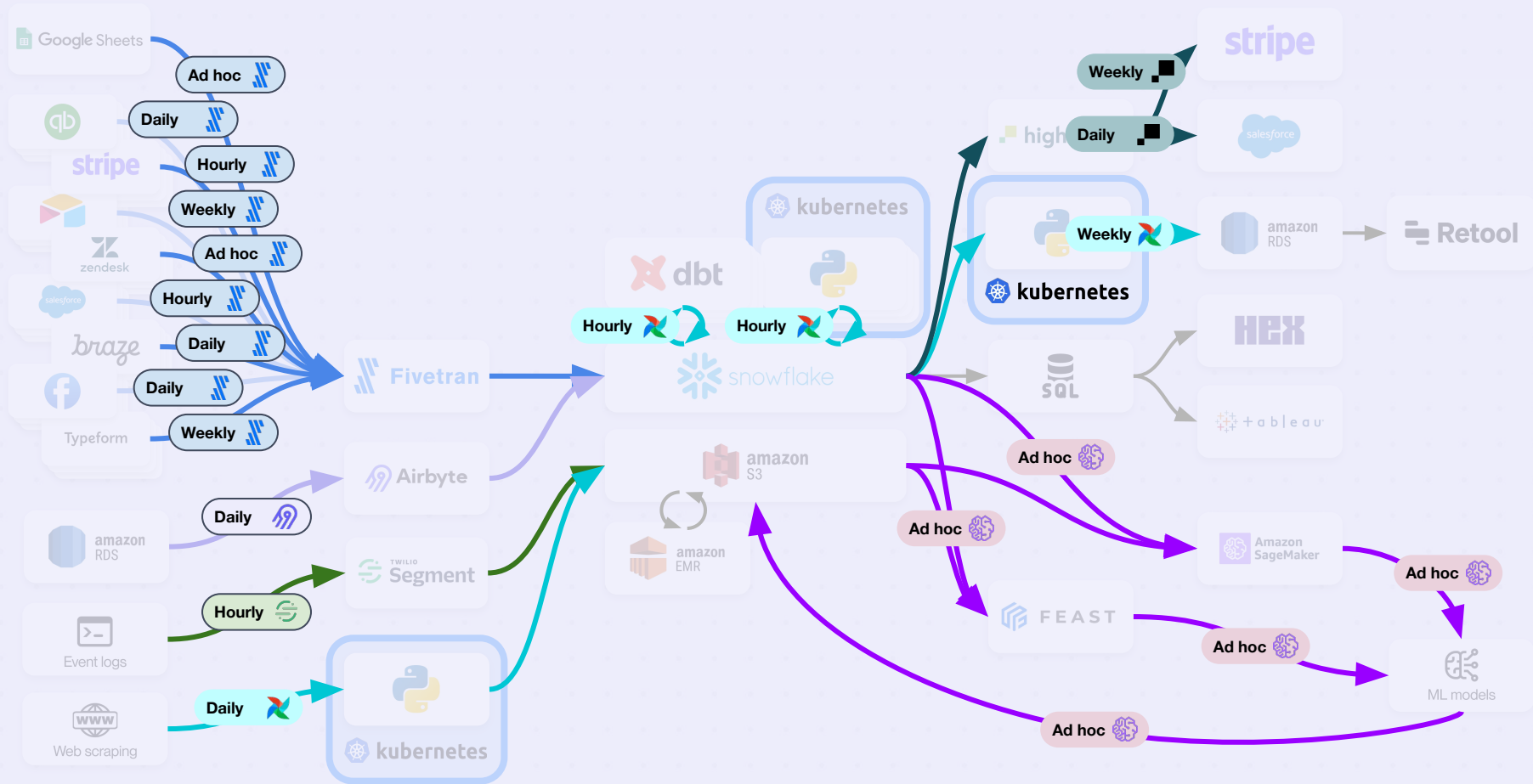


# Luke / Data Engineering Lead



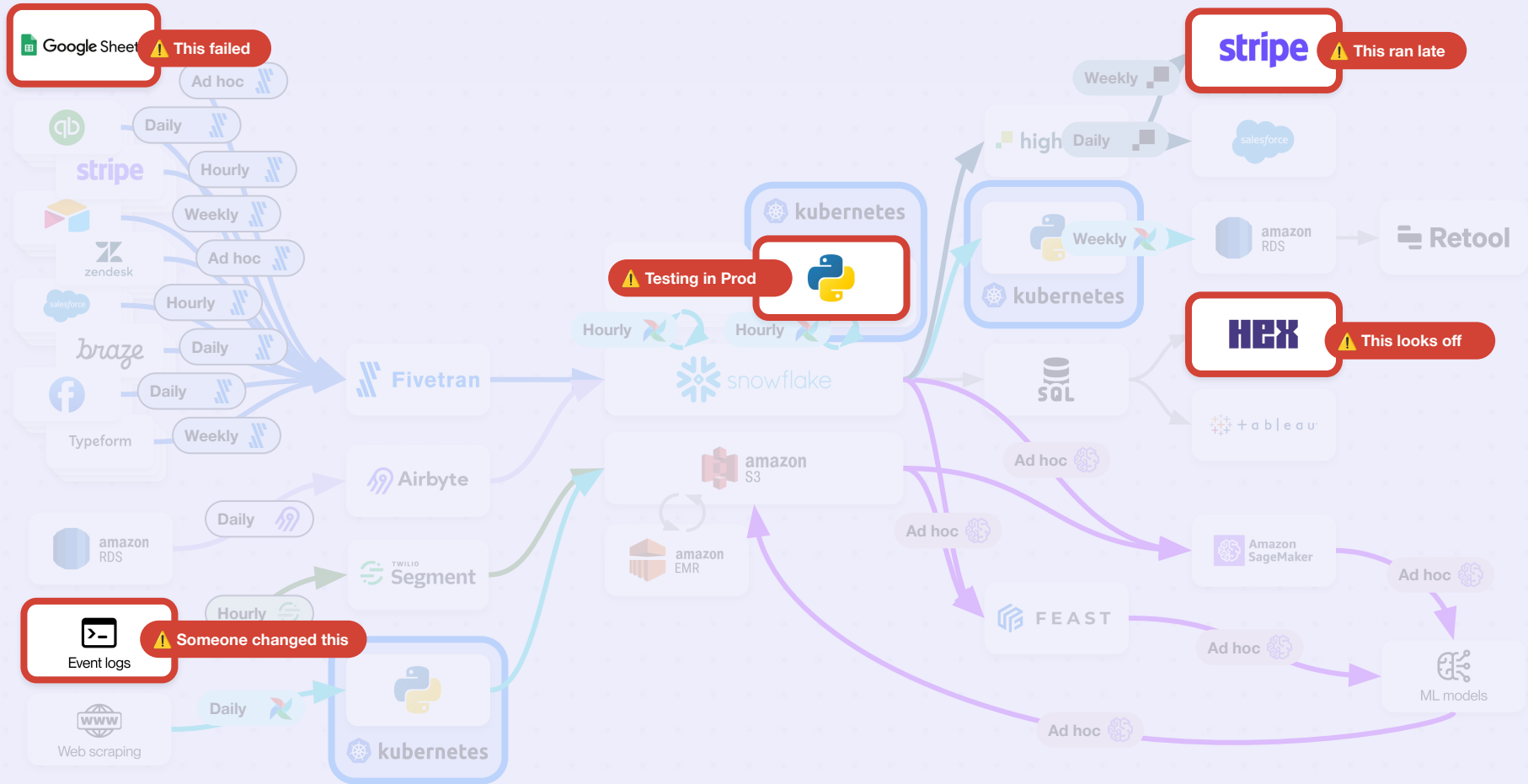


# Luke / Data Engineering Lead



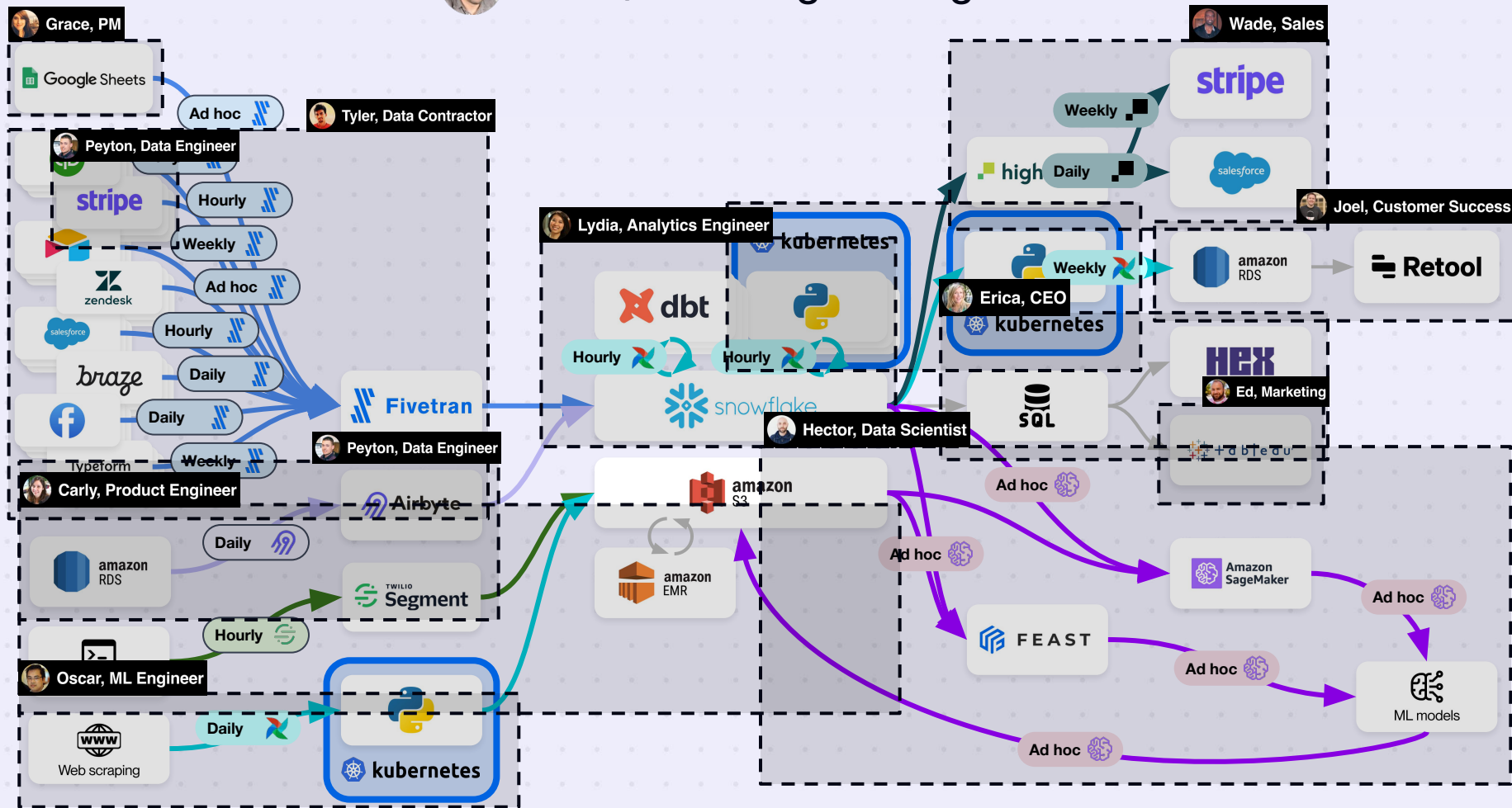


# Luke / Data Engineering Lead





# Luke / Data Engineering Lead





**Erica**

Why hasn't the OKR dashboard been updated today?



**Olivia**

This week's trial metrics look off. Can you investigate?



**Wade**

Who should I talk to about the lead scoring pipeline?



**Travis**

The Snowflake bill is 2x last month, what happened?



**Lydia**

My latest change just broke the marketing dashboard 😬. Can we revert?



**Grace**

Where is this data coming from?



**Hector**

Which 'Orders' table should I be using?

# Luke's life is chaos



## Slow & painful dev experience

Hard to identify, debug and fix problems across a large codebase that cuts across multiple tools



## Complex & costly tech stack

Dozens of point solutions to integrate and maintain with no “single pane of glass” for observability.



## Chaotic & intricate ownership

Difficult to strike a balance between centralization and decentralization while maximizing productivity.

**Why?**



# Skill issue?



Posted by u/[deleted] 3 months ago



539

## I Was Happier Being a Bartender Compared to Being a 6 Figure DE



Career



179 Comments



Share



Save



Hide



Report

Sort By: Best ▾

[View all comments](#)



iBortex · 3 mo. ago

Skill issue



1



Reply

Share



# ZIRPy venture-backed data companies?



**Lauren Balik** ✓ @laurenbalik · Sep 7, 2023



Warehouse-native is driven by VC funding incentives.

**Sequoia and Altimeter and to an extent ICONIQ have the Snowflake ecosystem.**

**Andreessen Horowitz and NEA own the Databricks ecosystem.**

Redpoint is closest to GCP **ecosystem**, but not really meaningful.

So **the** goal here for any...

[Show more](#)



**Matt Arderne** ✓ @mattarderne · Sep 7, 2023

General Data Observation:

current/new generation of data tooling is increasingly

- Use case specific
- Built on basis of a Data Warehouse ...

[Show more](#)

# Is data just *that* hard?



Check out the DataExpert.io Academy!



Login



## Zach Wilson's DataExpert.io Academy

An intermediate-level, live or self-paced, data engineering infrastructure and analytics engineering course and highly-motivated community!

Join the Full-access Live Boot Camp starting May 6th for \$2000

Join the Self-paced Data Engineering Course V4 Combined for \$1750

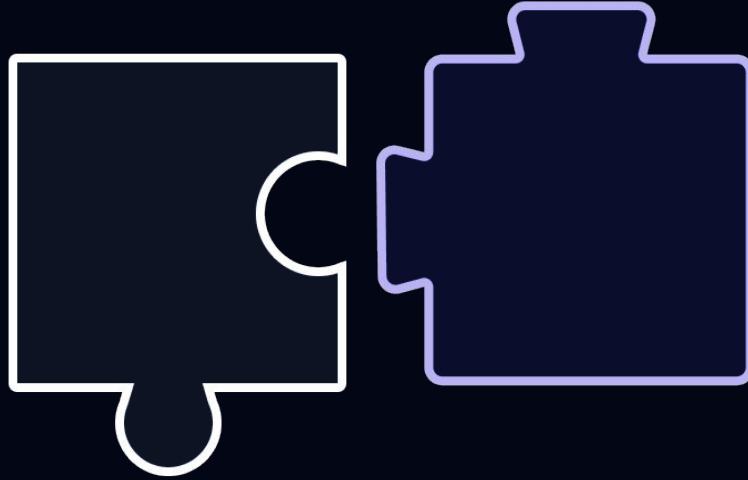
Sign Up

subscribe to my free newsletter

## The root technical cause

An **impedance mismatch** between workflow engines and the rest of the data platform

## Impedance mismatch?



When two layers of a system use  
fundamentally incompatible domain models

# Object–relational impedance mismatch

🌐 4 languages ▾

Article [Talk](#)

Tools ▾

From Wikipedia, the free encyclopedia

**Object–relational impedance mismatch** creates difficulties going from data in relational data stores ([relational database management system](#) [“RDBMS”]) to usage in domain-driven object models. Object-orientation (OO) is the default method for business-centric design in programming languages. The problem lies in neither relational nor OO, but in the conceptual difficulty mapping between the two logic models. Both are logical models implementable differently on database servers, programming languages, design patterns, or other technologies. Issues range from application to enterprise scale, whenever stored relational data is used in domain-driven object models, and vice versa. Object-oriented data stores can trade this problem for other implementation difficulties.

The term *impedance mismatch* comes from [impedance matching](#) in [electrical engineering](#) .

# The impedance mismatch in data

## Workflow Engines

### Workflow-oriented tools

Focused on the **task**: a function that performs some work and can depend on other tasks.

Examples: bash scripts, Python functions, K8s jobs



Airflow



Prefect



Github Actions

## The rest of the platform

### Asset-oriented tools

Focused on the **data asset**: an object in persistent storage that captures some understanding of the world

Examples: database tables, ML model, dashboards



dbt



Dagster



Fivetran



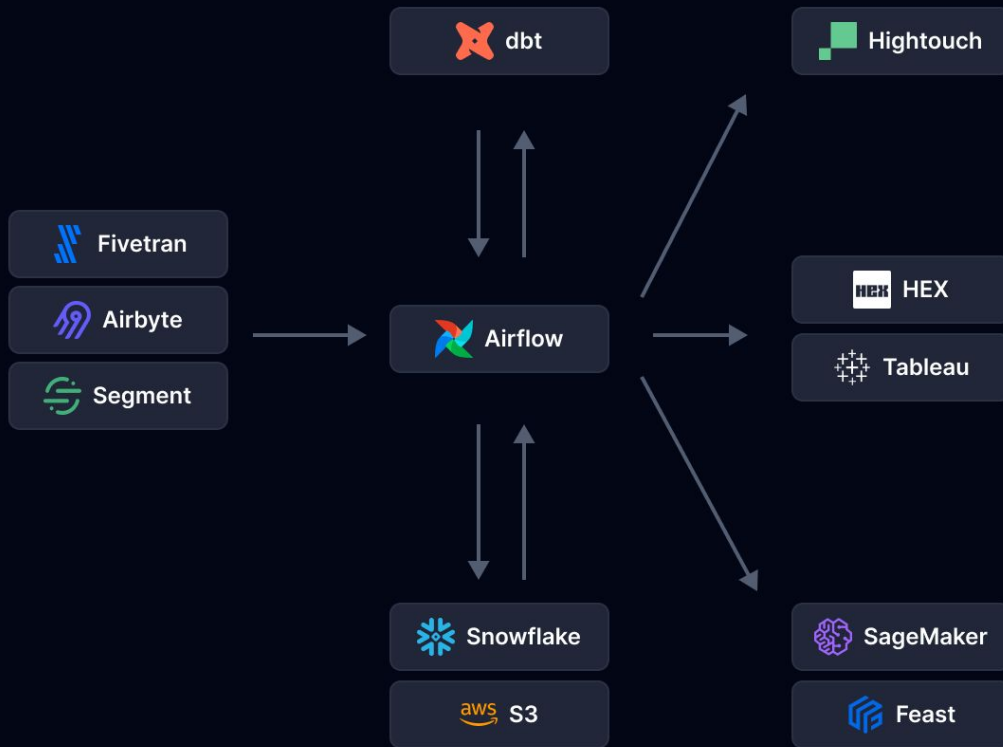
Snowflake



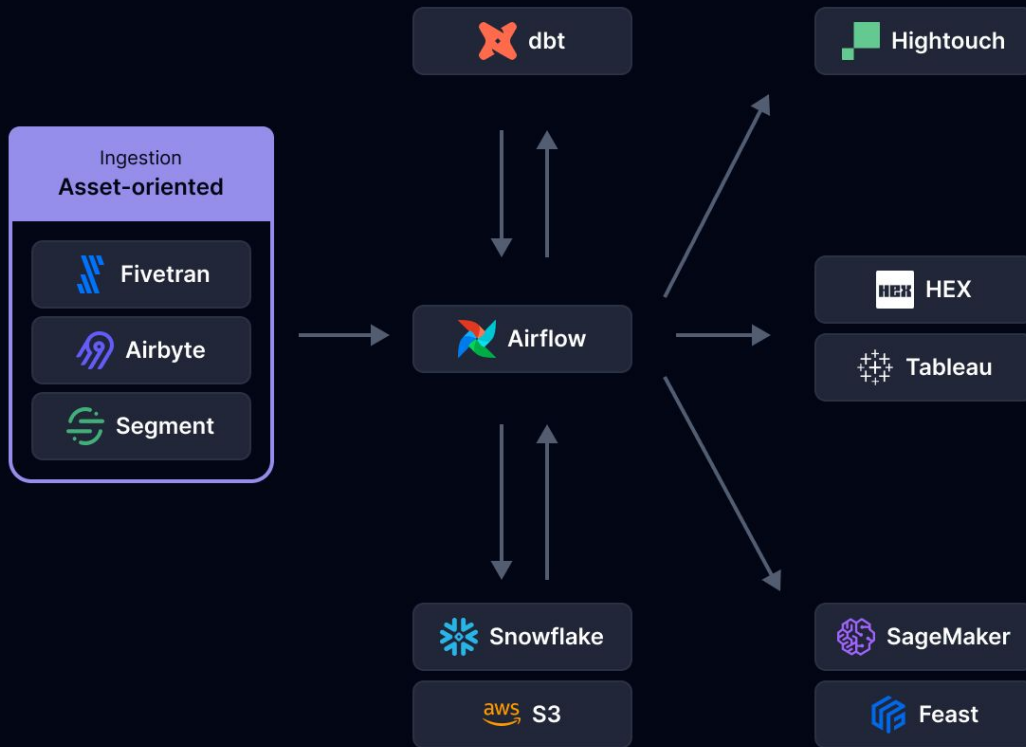
Airbyte

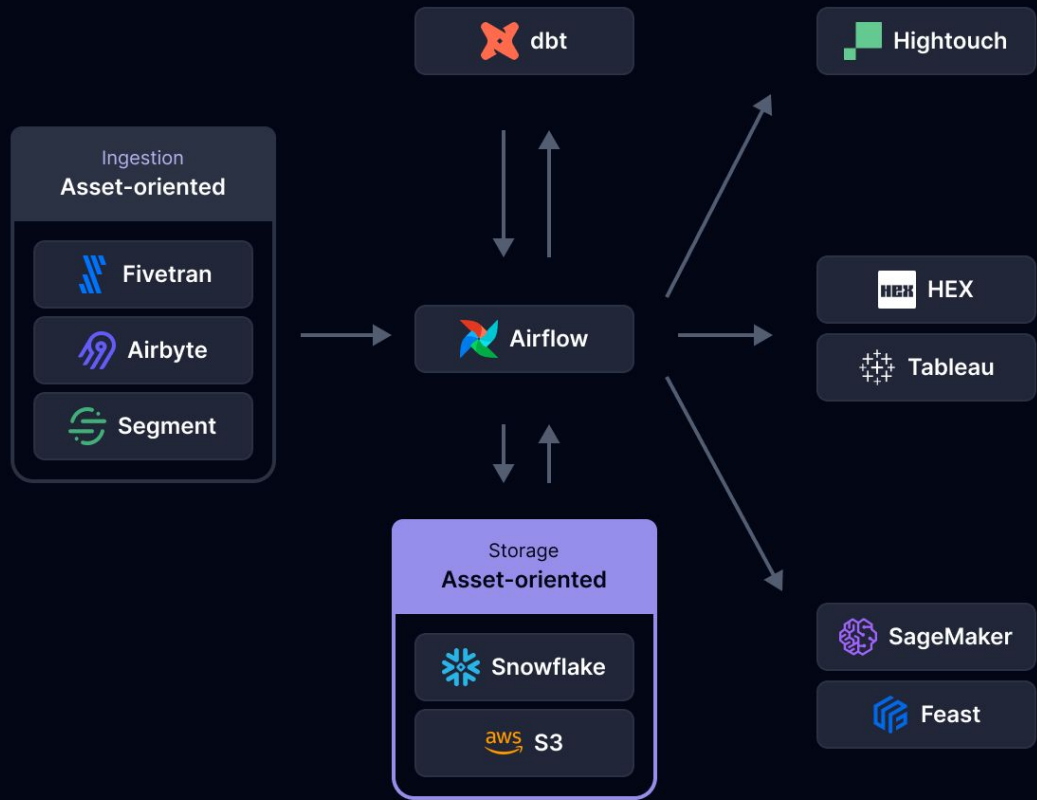


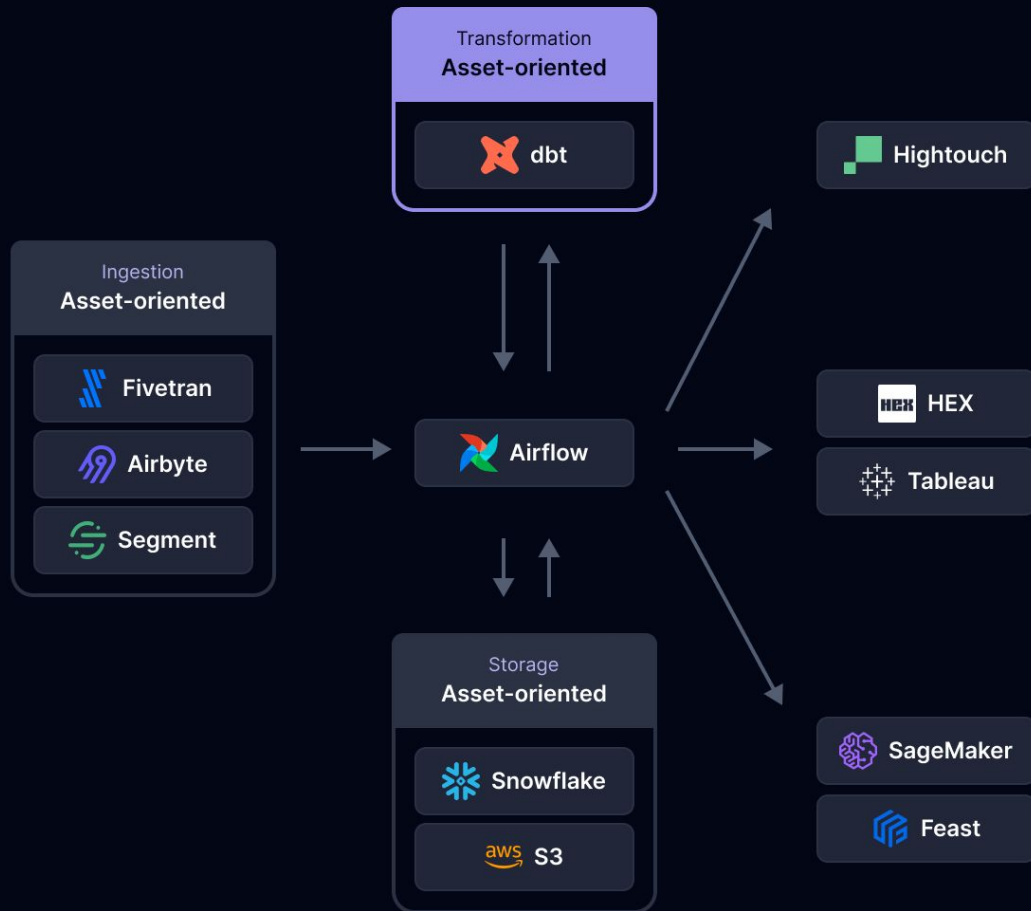
Monte Carlo

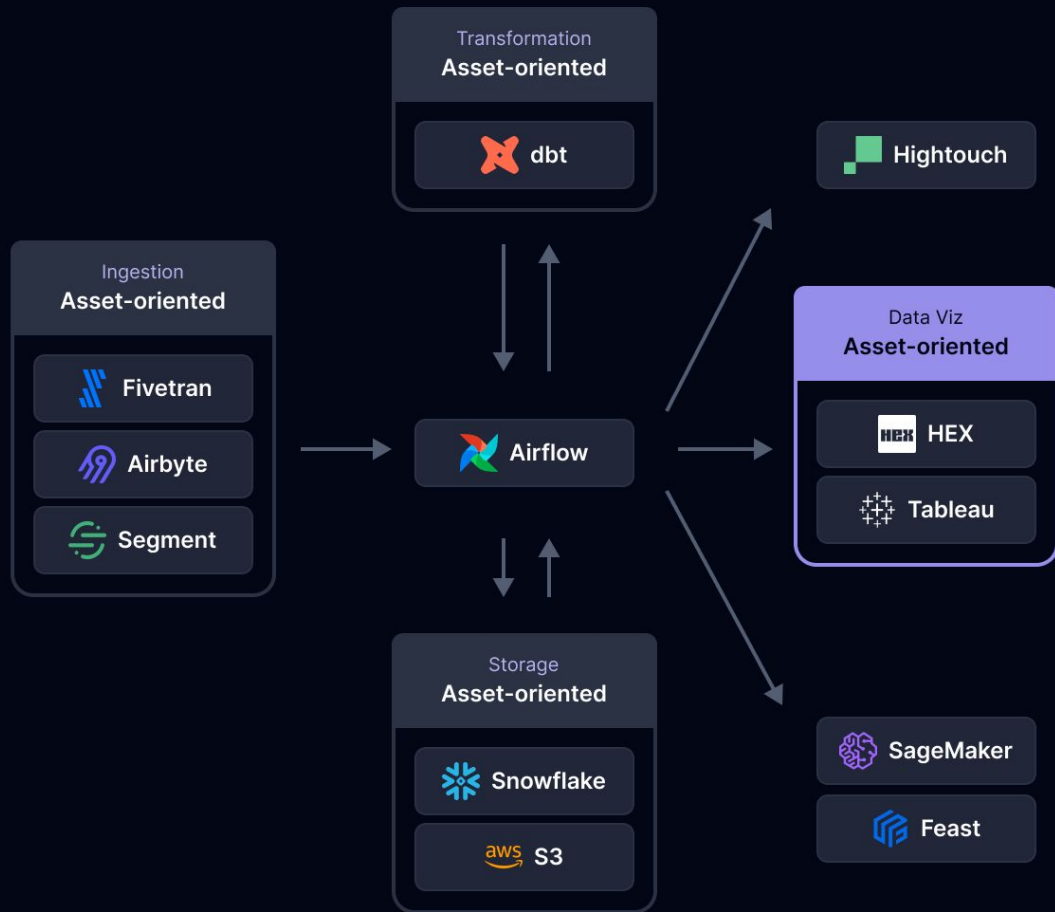


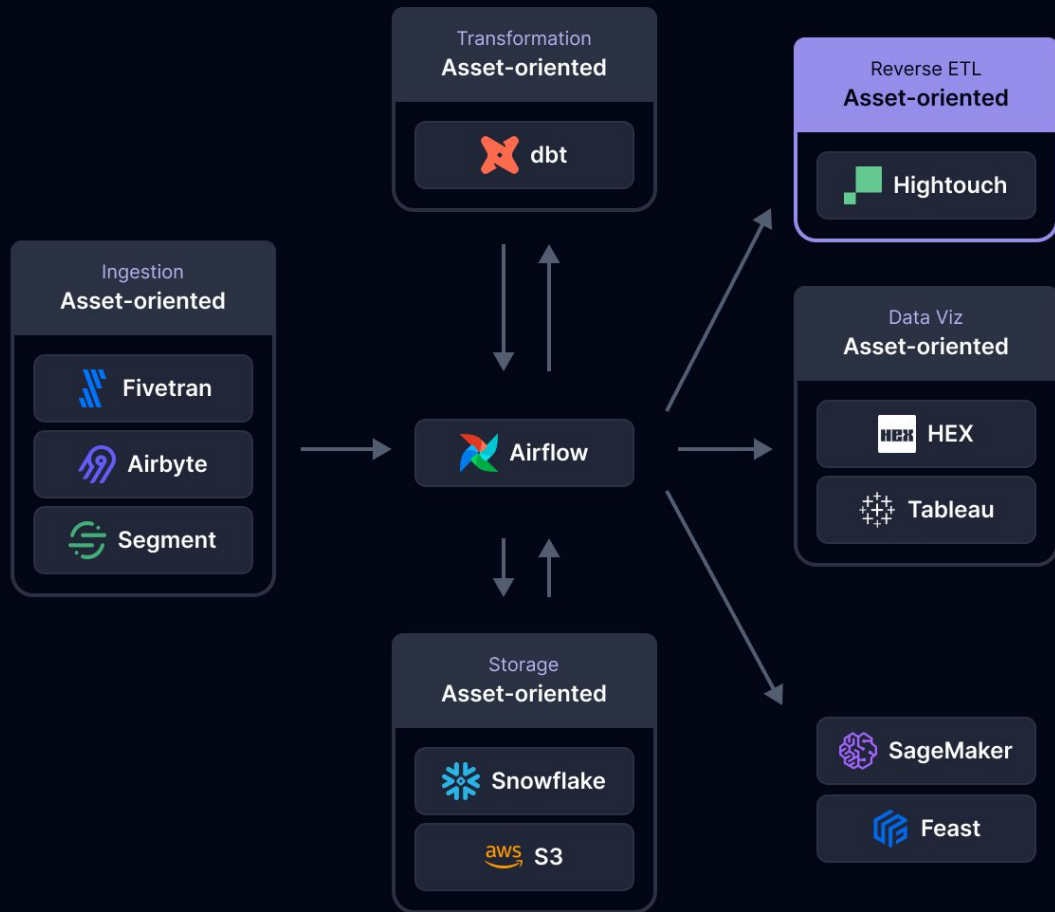


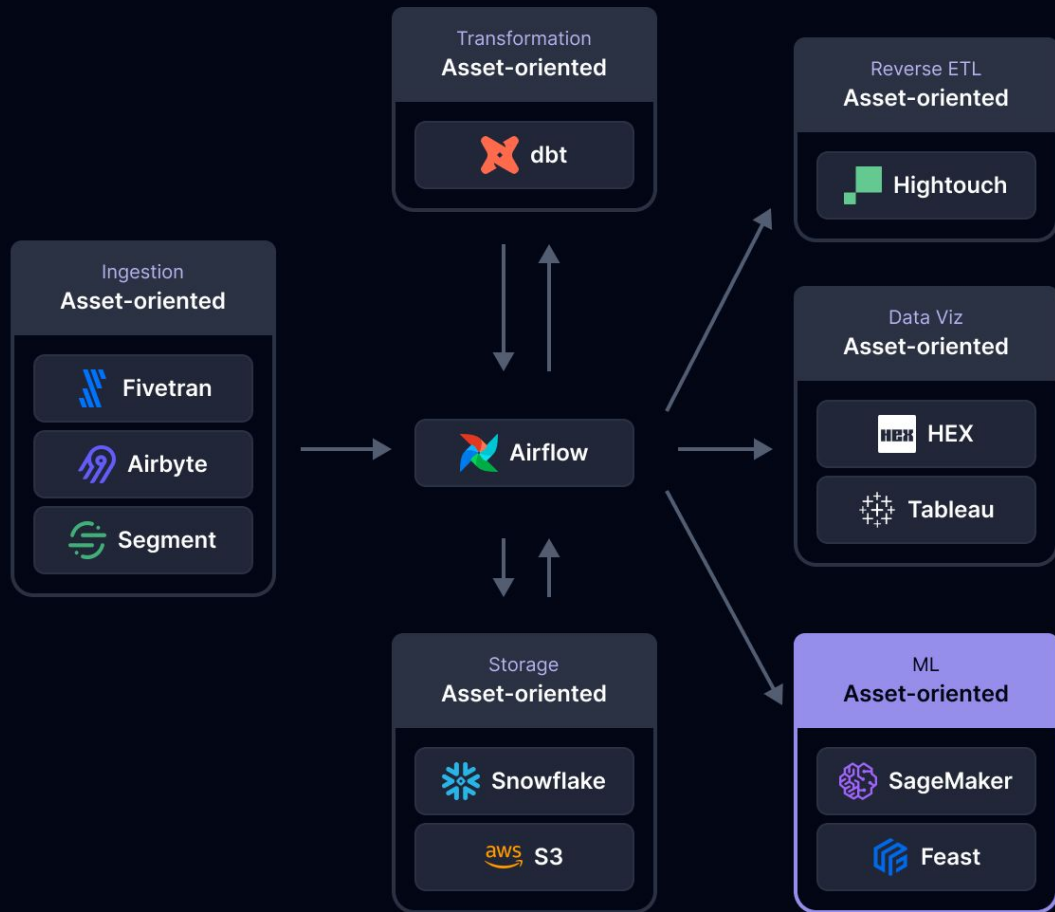














# What does this mean?

## Mismatched programming model

Engineers are more productive when they can think declaratively in terms of desired outputs (assets).

## Scheduling inflexibility

Workflow tasks do not have a notion of “freshness” leading to unnecessary spend by excessively rematerializing assets..

## Many-to-one relationship between assets and workflow steps

Single workflow tasks like ``dbt run`` may produce many data assets.

## Metadata and observability

The workflow engine has the execution logs, some other system has the schema. A separate data catalog needs to be integrated to join the data together and make it useful for stakeholders



# How does the world change when we move from workflow-orientation to asset-orientation?

Dev experience

Stack complexity

Ownership

**The dev experience  
is slow & painful**

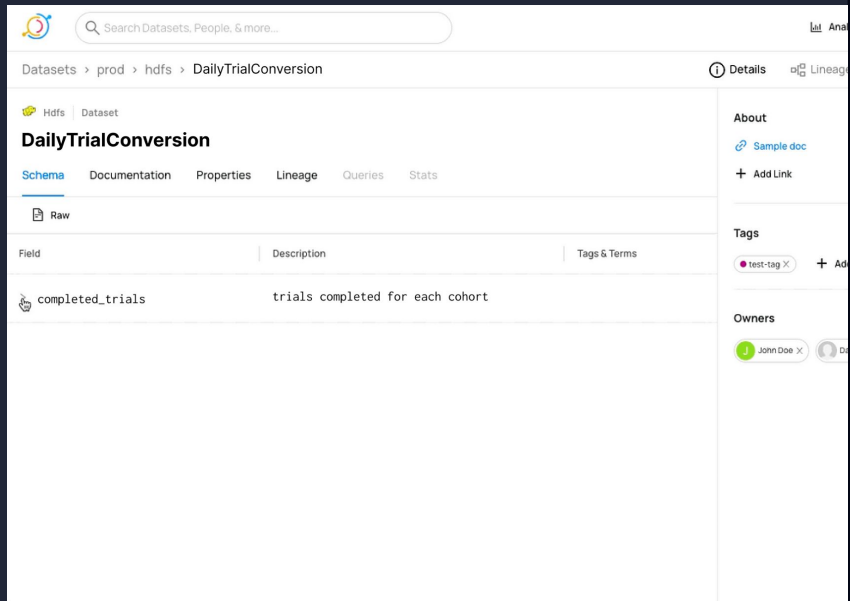
Our hero Luke gets a bug report that the weekly report is missing data.



**Olivia**

This week's trial metrics look off. Can you investigate?

He searches around his data catalog,  
but it was last updated 2 weeks ago  
because the sync job broke.



He has to spelunk through the codebase, reading git blame and grepping for every string he can think of to find the code that is related to the problematic data asset.

HelloWorld\_dag.py X

HelloWorld\_dag.py

```
1  from airflow import DAG
2  from airflow.operators.python import PythonOperator
3  from datetime import datetime
4
5  def helloWorld():
6      print("Hello World")
7
8  with DAG(dag_id="HelloWorld_dag",
9          start_date=datetime(2021,1,1),
10         schedule_interval="@hourly",
11         catchup=False) as dag:
12
13     task1 = PythonOperator(
14         task_id="hello_world",
15         python_callable=helloWorld)
16
17     task1
18
```

# The dev experience is slow & painful


He pushes it to the staging environment, which takes 15 minutes, to see if it works.



# The dev experience is slow & painful









Oops, he made a typo, time to wait another 15 minutes while we push again...

Update subscriptions pipeline #614

 Open lukeDev wants to merge 36 commits into `master` from `lk-fix-pipeline`

Conversation 2 Commits 36 Checks 1 Files changed 6

Commits on Feb 16, 2023

First commit	 f5bb18e <>
lukeDev committed 3 hours ago ❌	
Try again	 f5bb18e <>
lukeDev committed 3 hours ago ❌	
Why does this keep failing?	 f5bb18e <>
lukeDev committed 3 hours ago ❌	
WTF	 f5bb18e <>
lukeDev committed 3 hours ago ❌	
Try agiannnnnnnnn	 f5bb18e <>
lukeDev committed 3 hours ago ❌	
I think I got it	 f5bb18e <>
lukeDev committed 3 hours ago ❌	
OMGGGGGG	 f5bb18e <>
lukeDev committed 3 hours ago ❌	
Pleassseeee workkkkkkkk	 f5bb18e <>
lukeDev committed 3 hours ago ❌	

# How the impedance mismatch caused Luke's problems

## No single source of truth

- The data catalog had to assemble a view of the world using the exhaust of several tools
- Workflow-oriented orchestrator required manual integration to associate metadata with the entry in the catalog
- The integration between the two tools broke

## Opaque relationship between data assets and pipeline code

- Data pipelines were written in a workflow-oriented style.
- There was no clear correspondence between the data asset and the workflow task that produced it



# Workflow-oriented vs Asset-oriented code

```
extract_task = PythonOperator(
    task_id='extract_data',
    python_callable=extract_data,
    dag=dag,
)

transform_task = PythonOperator(
    task_id='transform_data',
    python_callable=transform_data,
    provide_context=True,
    dag=dag,
)

save_task = PythonOperator(
    task_id='save_to_s3',
    python_callable=save_to_s3,
    op_args=[ '{{ ti.xcom_pull(task_ids="transform_data") }}' ],
    op_kwargs={'execution_date': '{{ ts }}'},
    dag=dag,
)

extract_task > transform_task >> save_task
```

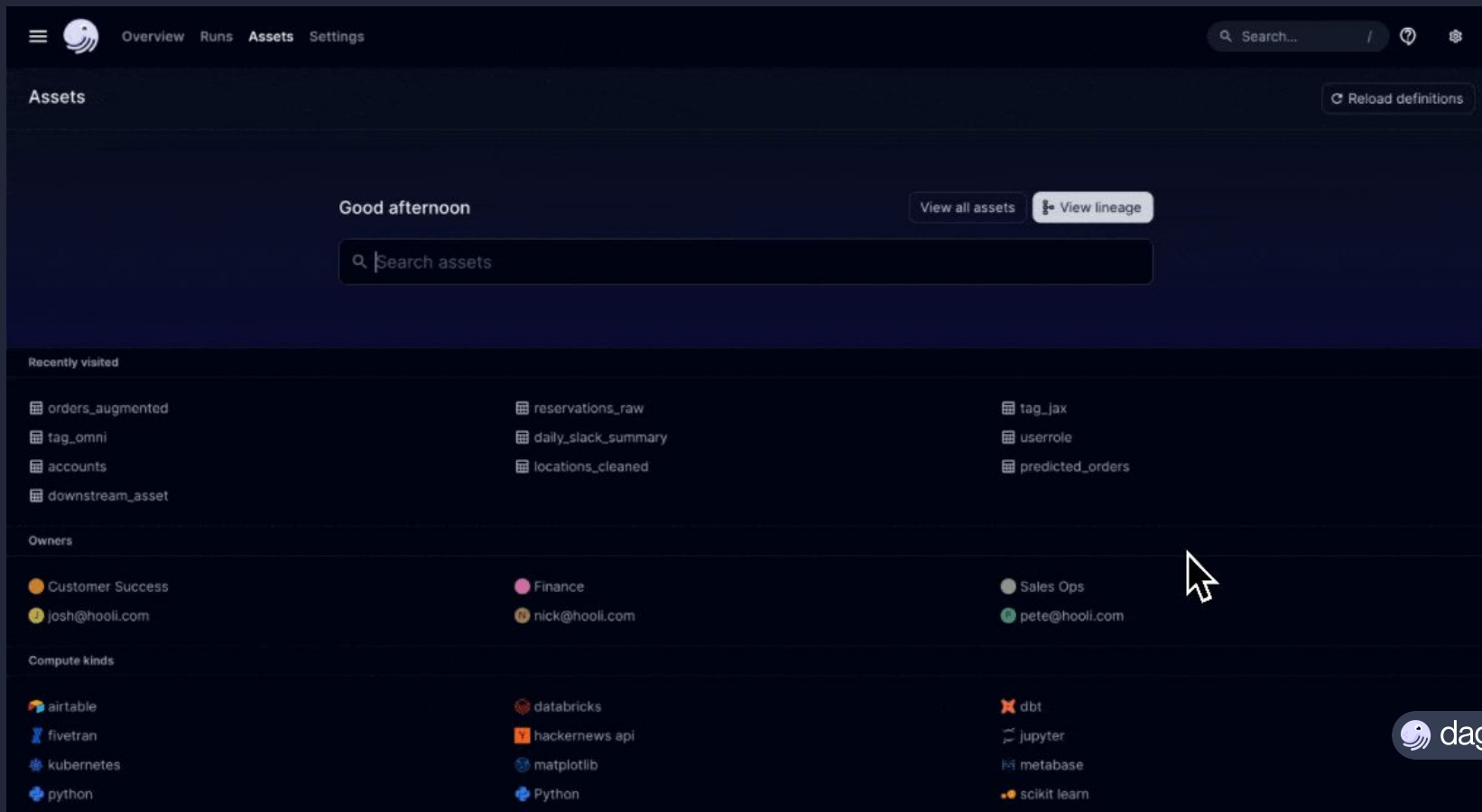
```
@asset
def clickstream_data():
    extract_data()
    transform_data()
    save_to_s3()
```

## Workflow-oriented UI

## DAGs

All 4	Active 2	Paused 2	Running 0	Failed 1	Filter DAGs by tag	Search DAGs	Auto-refresh	
DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks	Actions	Links
jaffle_shop	airflow	<div><div></div><div></div><div></div><div>1</div></div>	None	2024-03-17, 03:14:30		<div><div></div><div></div><div></div><div></div><div></div><div></div><div>1</div><div>3</div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>	<div>▶</div> <div>🗑️</div>	⋮
jaffle_shop_docker	airflow	<div><div></div><div></div><div></div><div></div></div>	None			<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>	<div>▶</div> <div>🗑️</div>	⋮
jaffle_shop_filtered	airflow	<div><div></div><div></div><div></div><div></div></div>	None			<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>	<div>▶</div> <div>🗑️</div>	⋮
jaffle_shop_kubernetes	airflow	<div><div></div><div></div><div></div><div></div></div>	1 day, 0:00:00		2024-03-24, 00:00:00	<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>	<div>▶</div> <div>🗑️</div>	⋮

# Asset-oriented UI - Catalog



The screenshot displays the Dagster Asset Catalog interface. At the top, a navigation bar includes a menu icon, the Dagster logo, and links for Overview, Runs, Assets, and Settings. A search bar is positioned on the right. Below the navigation bar, the 'Assets' section features a 'Good afternoon' greeting, a 'View all assets' button, a 'View lineage' button, and a search input field labeled 'Search assets'. The 'Recently visited' section lists assets such as orders\_augmented, tag\_omni, accounts, downstream\_asset, reservations\_raw, daily\_slack\_summary, locations\_cleaned, tag\_jax, userrole, and predicted\_orders. The 'Owners' section shows roles like Customer Success, Finance, and Sales Ops, each with associated email addresses. The 'Compute kinds' section lists various integrations including airtable, databricks, dbt, fivetran, hackernews api, jupyter, kubernetes, matplotlib, metabase, python, and scikit learn. A mouse cursor is visible over the Sales Ops role in the Owners section.

Assets

Good afternoon

View all assets View lineage

Search assets

Recently visited

- orders\_augmented
- tag\_omni
- accounts
- downstream\_asset
- reservations\_raw
- daily\_slack\_summary
- locations\_cleaned
- tag\_jax
- userrole
- predicted\_orders


Owners

- Customer Success
  - josh@hooli.com
- Finance
  - nick@hooli.com
- Sales Ops
  - pete@hooli.com

Compute kinds

- airtable
- fivetran
- kubernetes
- python
- databricks
- hackernews api
- matplotlib
- Python
- dbt
- jupyter
- metabase
- scikit learn

# Asset-oriented UI - Catalog

Overview Runs **Assets** Settings

Search... / ? ⚙️

Assets ↻ Reload definitions

Good afternoon View all assets 🔗 View lineage

🔍 Search assets

Recently visited


📊 orders_augmented	📊 reservations_raw	📊 tag_jax
📊 tag_omni	📊 daily_slack_summary	📊 userrole
📊 accounts	📊 locations_cleaned	📊 predicted_orders
📊 downstream_asset		

Owners

🟡 Customer Success	🟡 Finance	🟢 Sales Ops
👤 josh@hooli.com	👤 nick@hooli.com	👤 pete@hooli.com

Compute kinds

🎨 airtable	📦 databricks	🔴 dbt
🔵 fivetran	🟡 hackernews api	🔄 jupyter
🌐 kubernetes	📊 matplotlib	📊 metabase

 dagster +

# Asset-oriented UI - lineage

Global Asset Lineage

Jump to... Filter ✨ Type an asset subset... (ex: weekly\_order\_summary+) ⌚ ⌂ ↻ Materialize all

Asset groups is any of ANALYTICS, RAW\_DATA, CLEANED ✕

hooli

- ANALYTICS
  - company\_perf
  - company\_stats
  - order\_forecast\_model
  - order\_stats
  - orders\_augmented
  - sku\_stats
  - weekly\_order\_summary
- CLEANED
  - locations\_cleaned
  - orders\_cleaned
  - users\_cleaned
- RAW\_DATA
  - locations
  - orders
  - users

Diagram illustrating the Global Asset Lineage for the 'hooli' dataset, showing the flow from RAW\_DATA to CLEANED and then to ANALYTICS.

**RAW\_DATA** (Assets: locations, orders, users)

**CLEANED** (Assets: locations\_cleaned, orders\_cleaned, users\_cleaned)

**ANALYTICS** (Assets: company\_stats, company\_perf, order\_stats, orders\_augmented, order\_forecast\_model, weekly\_order\_summary, sku\_stats)

Lineage Flow:

- locations → locations\_cleaned → orders\_augmented
- orders → orders\_cleaned → order\_stats
- users → users\_cleaned → order\_stats
- orders\_augmented → order\_stats
- order\_stats → weekly\_order\_summary
- order\_stats → sku\_stats
- order\_stats → order\_forecast\_model
- order\_stats → company\_stats
- order\_stats → company\_perf

Additional context: Asset groups is any of ANALYTICS, RAW\_DATA, CLEANED ✕

# Asset-oriented UI - lineage

The screenshot displays the Dagster Assets interface. At the top, a navigation bar includes a menu icon, the Dagster logo, and links for Overview, Runs, Assets (selected), and Settings. A search bar and utility icons are on the right. The main header shows 'Assets' and a 'Reload definitions' button. A greeting 'Good afternoon' is followed by 'View all assets' and 'View lineage' buttons. A large search bar is labeled 'Search assets'. Below, three sections are visible: 'Recently visited' with a grid of asset names (orders\_augmented, tag\_omni, accounts, downstream\_asset, reservations\_raw, daily\_slack\_summary, locations\_cleaned, tag\_jax, userrole, predicted\_orders); 'Owners' with a grid of roles and emails (Customer Success, josh@hooli.com, Finance, nick@hooli.com, Sales Ops, pete@hooli.com); and 'Compute kinds' with a grid of technologies (airtable, databricks, dbt, fivetran, hackernews api, jupyter, kubernetes, matplotlib, metabase, python, Python, scikit learn). A mouse cursor points to the Finance owner card.

Assets Reload definitions

Good afternoon View all assets View lineage

Search assets

Recently visited

- orders\_augmented
- tag\_omni
- accounts
- downstream\_asset
- reservations\_raw
- daily\_slack\_summary
- locations\_cleaned
- tag\_jax
- userrole
- predicted\_orders

Owners

- Customer Success
- josh@hooli.com
- Finance
- nick@hooli.com
- Sales Ops
- pete@hooli.com

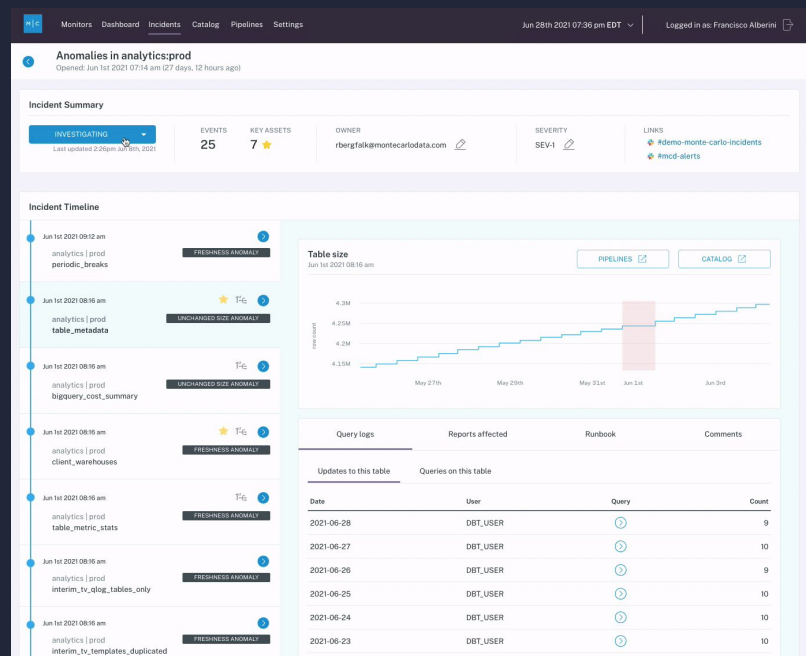
Compute kinds

- airtable
- fivetran
- kubernetes
- python
- databricks
- hackernews api
- matplotlib
- Python
- dbt
- jupyter
- metabase
- scikit learn

**The tech stack is  
complex & expensive**

Luke's customers, data pipeline authors, want to be alerted if their data fails quality checks.

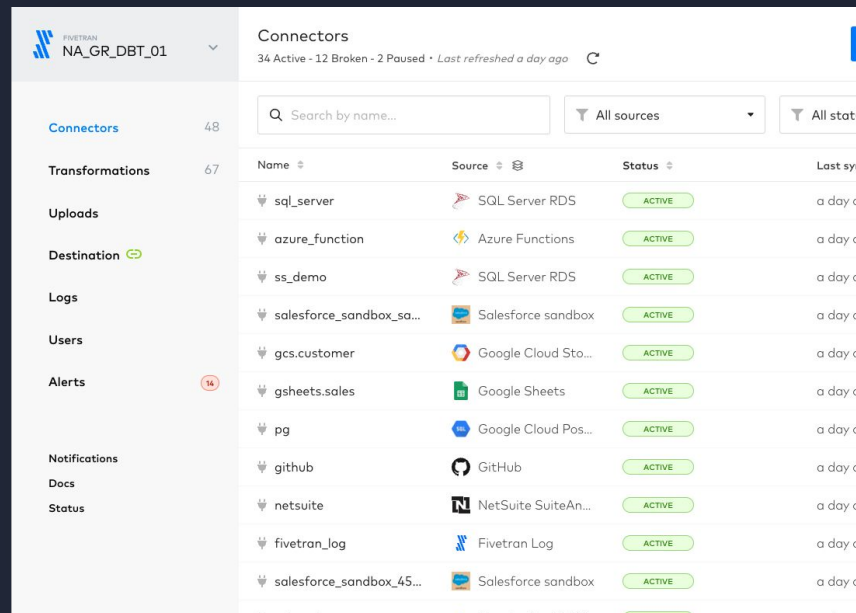
- He negotiates a deal with a vendor
- Asks his stakeholders to tag all of their queries
- They say: "Q1 2025"





Luke's customers also want to move some data around.

- He negotiates a deal with an ELT vendor
- Writes a custom operator for his stakeholders to use
- Needs to remember to wire it up to every other tool in the stack



The screenshot shows the Fivetran Connectors page for workspace NA\_GR\_DBT\_01. It displays a list of 14 active connectors, each with a name, source, status, and last sync time. The connectors include sql\_server, azure\_function, ss\_demo, salesforce\_sandbox\_sa..., gcs.customer, gsheets.sales, pg, github, netsuite, fivetran\_log, and salesforce\_sandbox\_45....

Name	Source	Status	Last sync
sql_server	SQL Server RDS	ACTIVE	a day ago
azure_function	Azure Functions	ACTIVE	a day ago
ss_demo	SQL Server RDS	ACTIVE	a day ago
salesforce_sandbox_sa...	Salesforce sandbox	ACTIVE	a day ago
gcs.customer	Google Cloud Sto...	ACTIVE	a day ago
gsheets.sales	Google Sheets	ACTIVE	a day ago
pg	Google Cloud Pos...	ACTIVE	a day ago
github	GitHub	ACTIVE	a day ago
netsuite	NetSuite SuiteAn...	ACTIVE	a day ago
fivetran_log	Fivetran Log	ACTIVE	a day ago
salesforce_sandbox_45...	Salesforce sandbox	ACTIVE	a day ago

## Luke gets the Snowflake bill for the month and it's up 400%

- He looks at the query\_history table and sees its driven by a query that has no attribution information
- He greps through the codebase and eventually finds the code that issues the query
- He fixes the bug, but asks his customers to tag their queries for next time.
- They say: "Q1 2025"



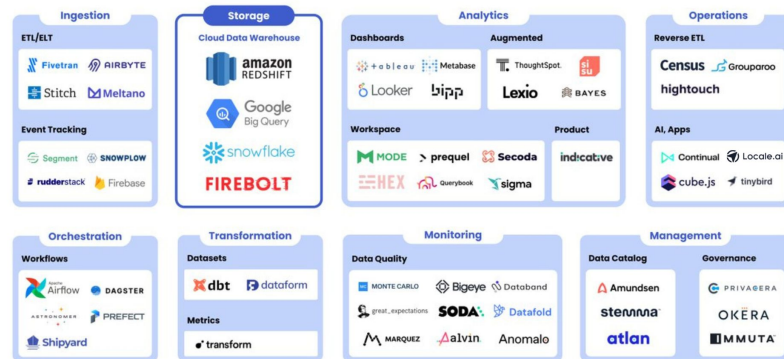
**Travis**

The Snowflake bill is 4x last month, what happened?

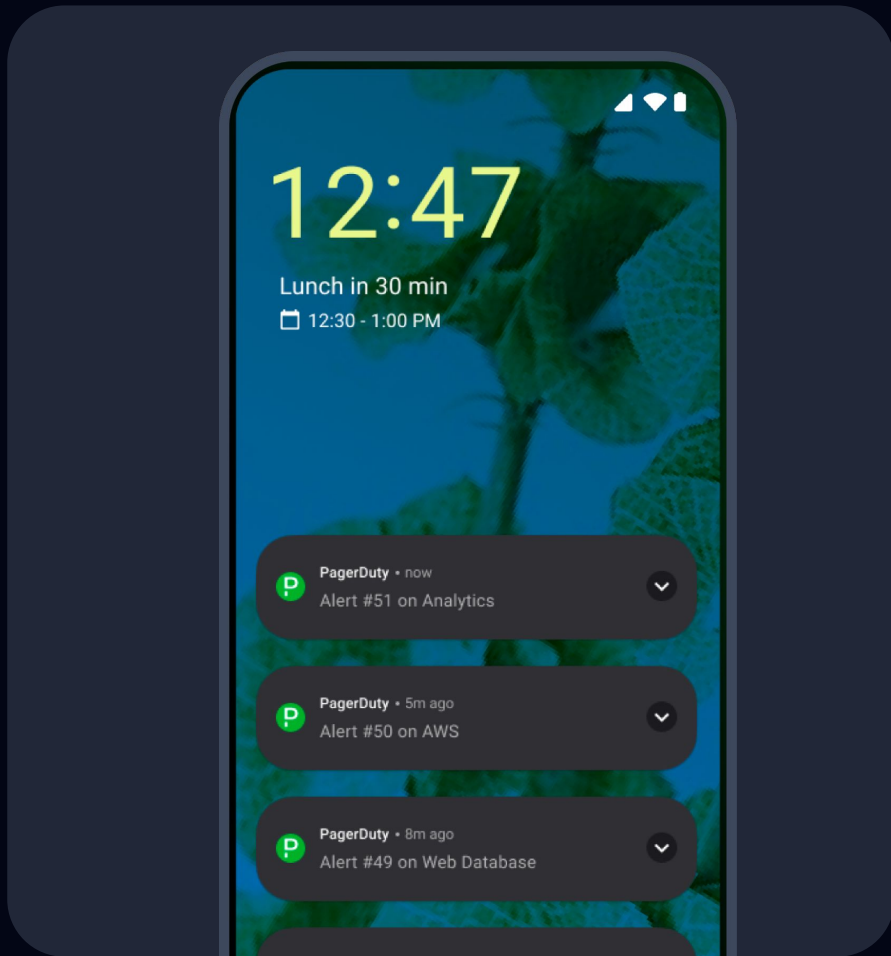
By the end of the year, he's bought 20 different data tools and runs 10 new OSS services.

## The Modern Data Stack

@ValentinUmbach  
2021-07-22



Oh, and his platform eng counterparts just got laid off, so he has to carry the pager every week now.



# Why the impedance mismatch made Luke buy more tools

## Disconnected systems

Because Fivetran and Monte Carlo are asset-oriented tools and Airflow is not, Luke had to prod his stakeholders to do an expensive, manual integration phase for each data asset.

## Manual integration

Similarly, he was unable to attribute Snowflake spend to specific teams or assets without hours or days of sifting through the codebase, or asking his stakeholders to, again, manually tag each one of their queries.

## Poor observability

Because Airflow is a workflow engine, it does not have any built-in features that support asset-oriented capabilities like data observability, cost management and data discovery, necessitating the purchase, integration and maintenance of numerous point solutions.

# Why the impedance mismatch made Luke buy more tools

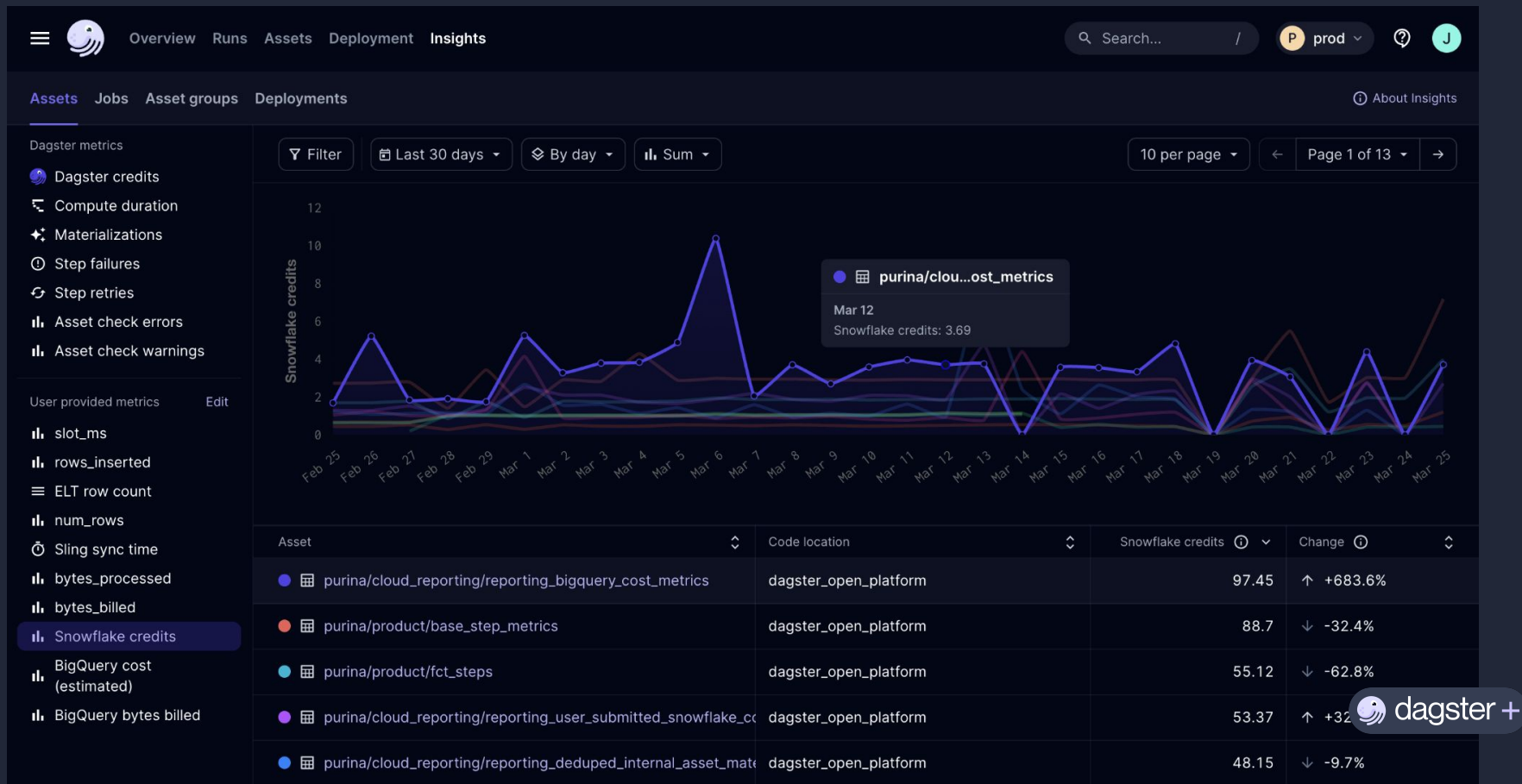
## Manual tagging of queries

- Airflow is workflow-oriented, and has no knowledge of which tasks correspond to which data assets
- For this reason, Airflow can't help Luke's customers with this problem, so they must tag their queries with asset attribution manually, which is expensive and fragile.

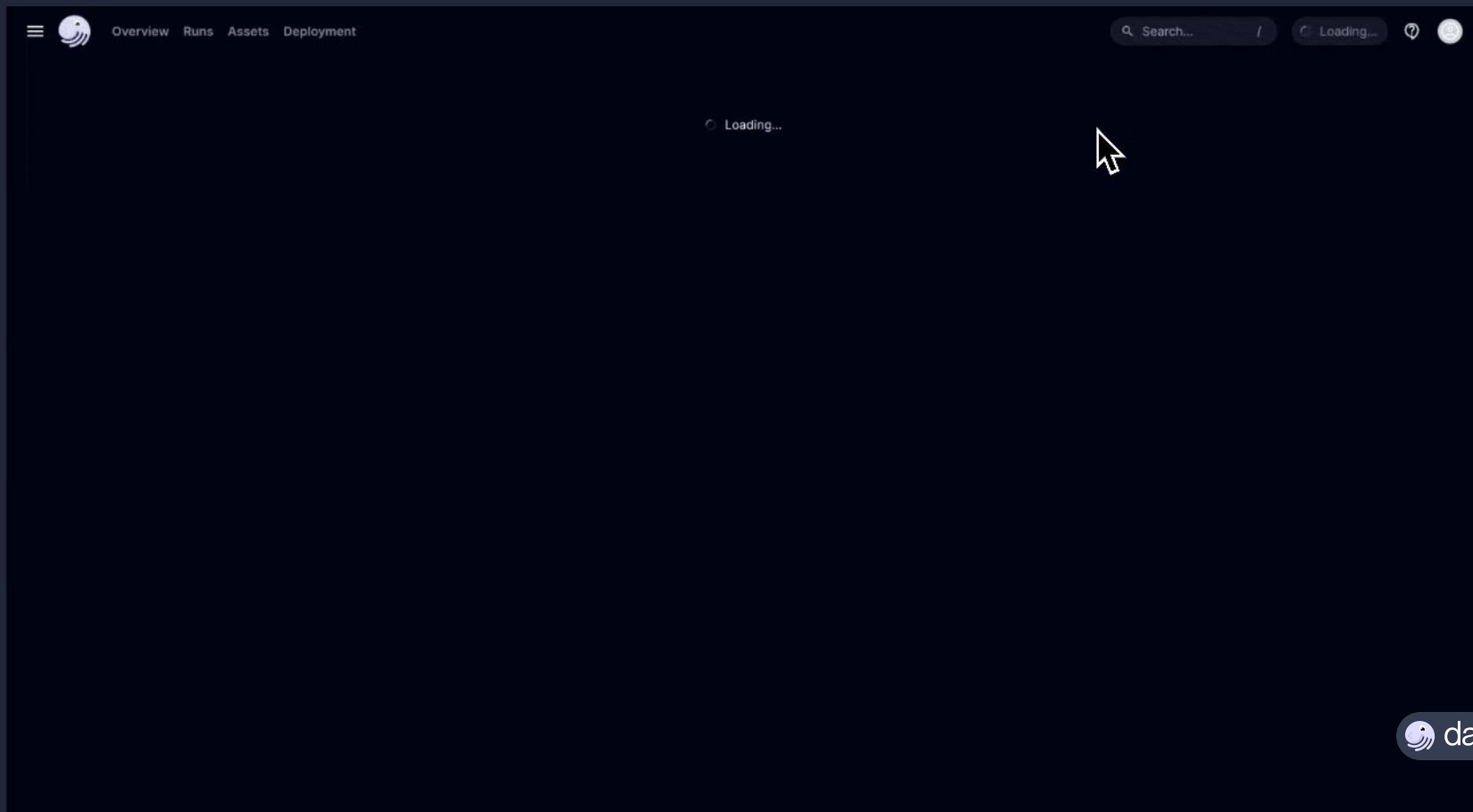
## No data observability

- As tasks in workflow engines are black boxes, Airflow has no knowledge of the data they are operating on.
- Thus, Airflow's observability is limited to high-level cluster and workflow health.
- There is no visibility into the data itself, necessitating manual integration of point solutions.

# Cost control with Dagster Insights




# Cost control with Dagster Insights





# Data Quality checks with Dagster


 Overview Runs **Catalog** Settings Insights


Catalog > All assets > test\_prefix / checked\_asset


Overview Events **Checks** Plots Lineage Insights


Checks (4) Execute all

Filter checks

 always\_fail  
Failed

 random\_fail\_check  
Failed

 severe\_random\_fail\_check  
Failed


 slow\_check  
Succeeded

☒ random\_fail\_check Execute

About

A check that fails half the time.


Latest execution

Evaluation result	Timestamp	Target materialization
 Failed	Jan 26, 7:08 PM	Dec 6, 2023, 5:50 PM

Metadata

timestamp 1706314208.0814714




Execution history


Evaluation result	Timestamp	Target materialization	Metadata
 Passed	Dec 6, 2023, 5:07 PM	Dec 6, 2023, 5:06 PM	timestamp 1701900424.3014219

checked\_asset


No description

Materialized Dec 6, 2023, 5:50 PM

Checks  2  1  1

 dagster +

# Data Quality checks with Dagster



OverviewRunsAssetsSettingsInsights


Assets > test\_prefix / checked\_asset


OverviewEventsChecksPlotsLineageInsights


Checks (4)


Execute all

Filter checks

 always\_fail  
Failed

 random\_fail\_check  
Failed

 severe\_random\_fail\_check  
Failed

 slow\_check  
Succeeded


always\_fail

Execute

About

A check that always fails, and has several types of metadata.





Latest execution

Evaluation result	Timestamp	Target materialization
 Failed	Mar 7, 1:51 PM	Dec 6, 2023, 5:50 PM

Metadata

asset_key	test_prefix / checked_asset
foo	bar




Execution history

Evaluation result	Timestamp	Target materialization	Metadata
 Failed	Jan 26, 7:10 PM	Dec 6, 2023, 5:50 PM	<div>View metadata</div>
 Failed	Dec 6, 2023, 5:07 PM	Dec 6, 2023, 5:08 PM	<div>View metadata</div>
 Failed	Sep 7, 2023, 11:21 AM	Sep 7, 2023, 11:21 AM	<div>View metadata</div>
 Failed	Sep 7, 2023, 11:19 AM	Sep 7, 2023, 11:18 AM	<div>View metadata</div>

checked\_asset

No description

MaterializedDec 6, 2023, 5:50 PM

Checks 2 1 1

 dagster +

**Ownership is chaotic  
& complicated**

All of the data teams got reorged,  
and now workflows are shared  
between multiple teams.



**Olivia**

**Organization update**

I'm excited to share some changes to our engineering org...

Teams start to step on each others' toes technically and socially.



They eventually conclude that it's easier to spin up separate infrastructure for their workflows than collaborate on a single instance.



Because there are multiple owners and Airflow instances, stakeholders are more confused than ever. The [#help-data](#) Slack channel is chaos.



**Wade**

Who should I talk to about the lead scoring pipeline?



**Grace**

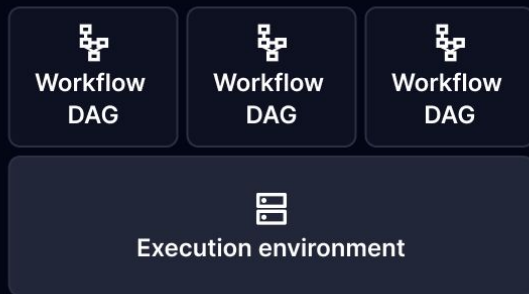
Where is this data coming from?



**Hector**

Which 'Orders' table should I be using?

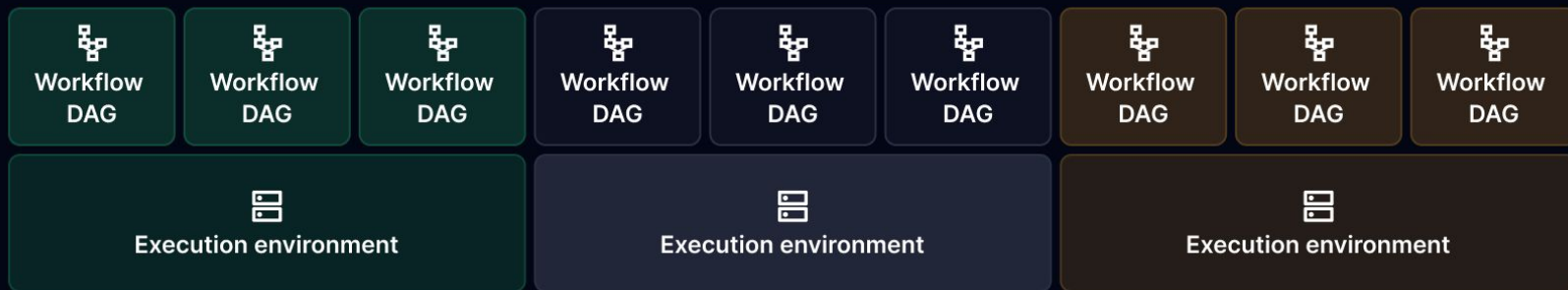
# Why the impedance mismatch made ownership complicated



Workflow-oriented orchestration (Airflow, others)



# Why the impedance mismatch made ownership complicated



Workflow-oriented orchestration (Airflow, others)

# Why the impedance mismatch made ownership complicated




Asset-oriented orchestration adds an overlay layer

# Wrapping up

- There is an impedance mismatch between **workflow-oriented** orchestrators and the rest of the (**asset-oriented**) data platform.
- This impedance mismatch causes problems with:
  - Developer experience
  - Stack complexity
  - Collaboration
- There is a way out

# Thanks!

- @floydophone on Twitter
- [linkedin.com/in/pwhunt](https://www.linkedin.com/in/pwhunt) on LinkedIn
- [dagster.io](https://dagster.io) for Dagster
-  dagster +