dagster

Orchestrating Software-Defined Assets

Sandy Ryza (@s_ryz)
Lead Engineer, Dagster Project - Elementl

👋 - Sandy

Building tools for data people

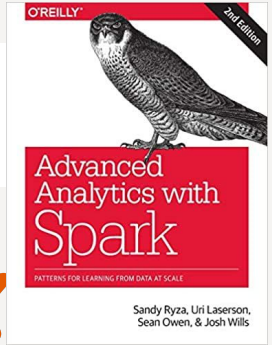
Being a data person

Building tools for data people

cloudera

hadoop

APACHE Spark™

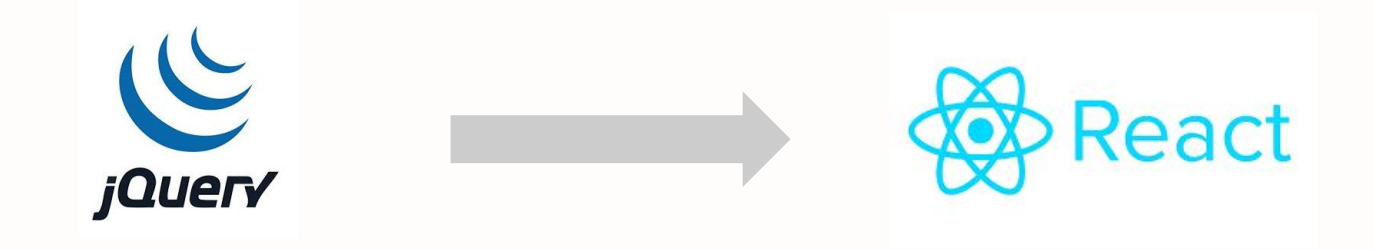


Clover Health

dagster

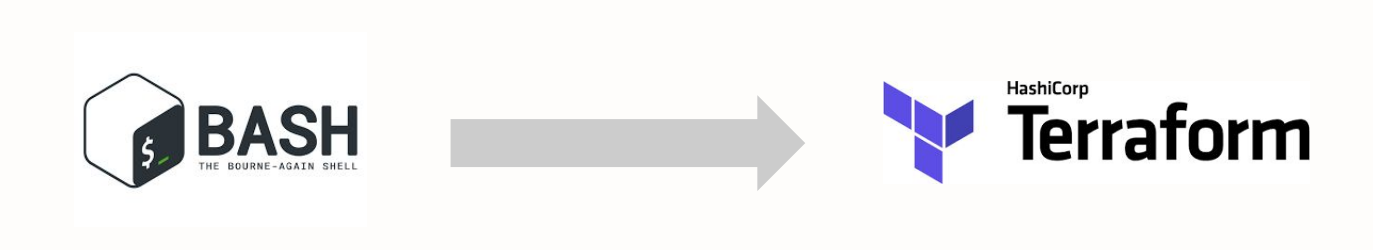
KEEP TRUCKIN

Frontend



Cluster Orchestration

Dev Ops



Imperative → Declarative

Imperative

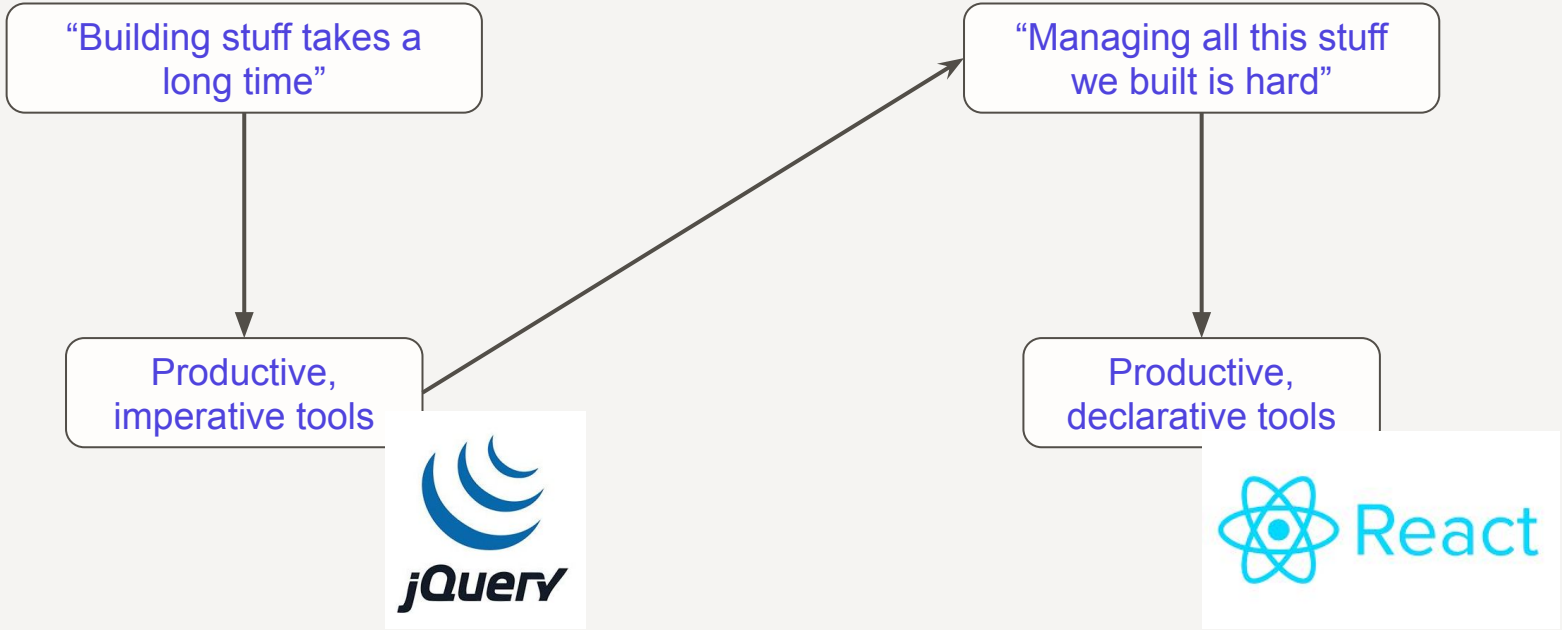
You give commands telling your system what to do.

Declarative

You describe the end state you want your system to be in.

What is going on here?

The software domain trajectory

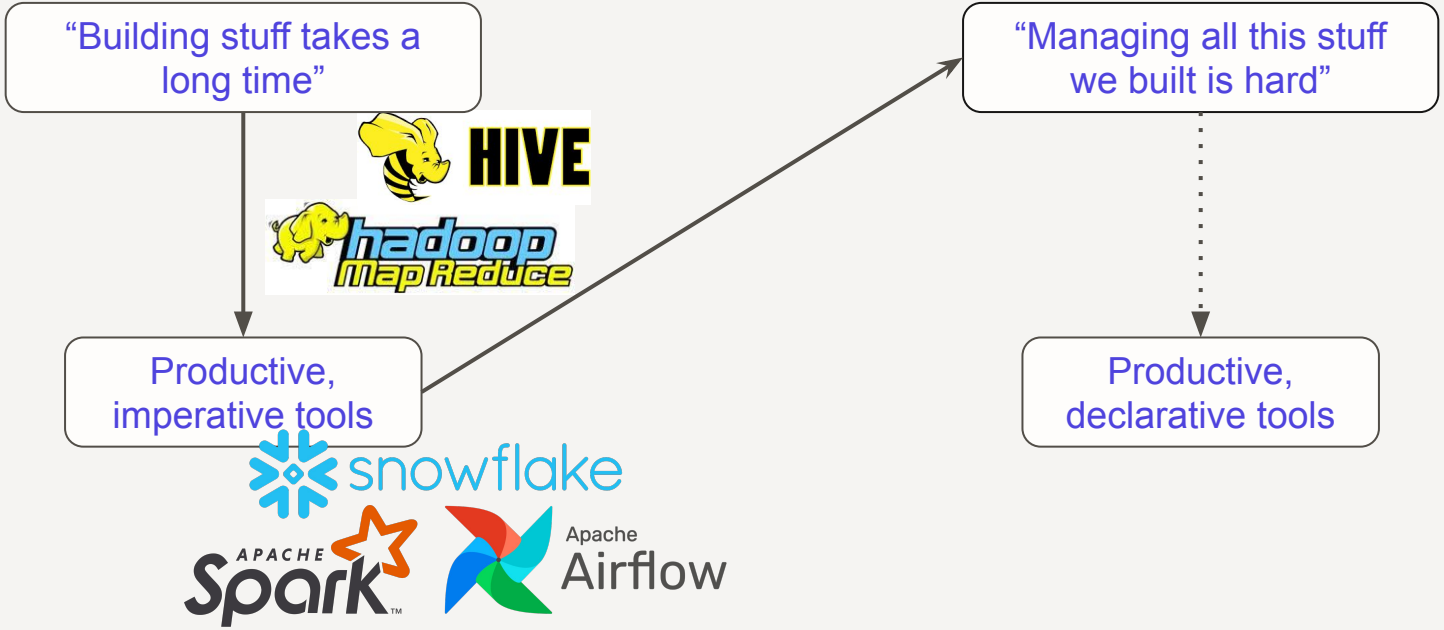


Declarative tools stop complexity from turning into chaos

- Make intentions explicit
- Offer a principled way of managing change

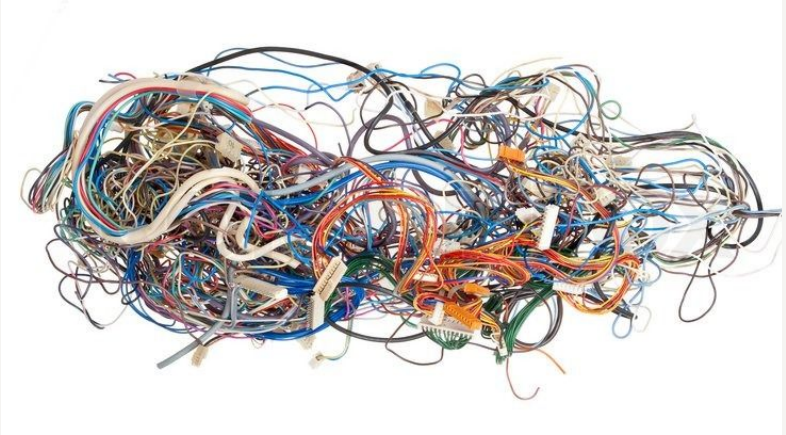
DATA

The data trajectory

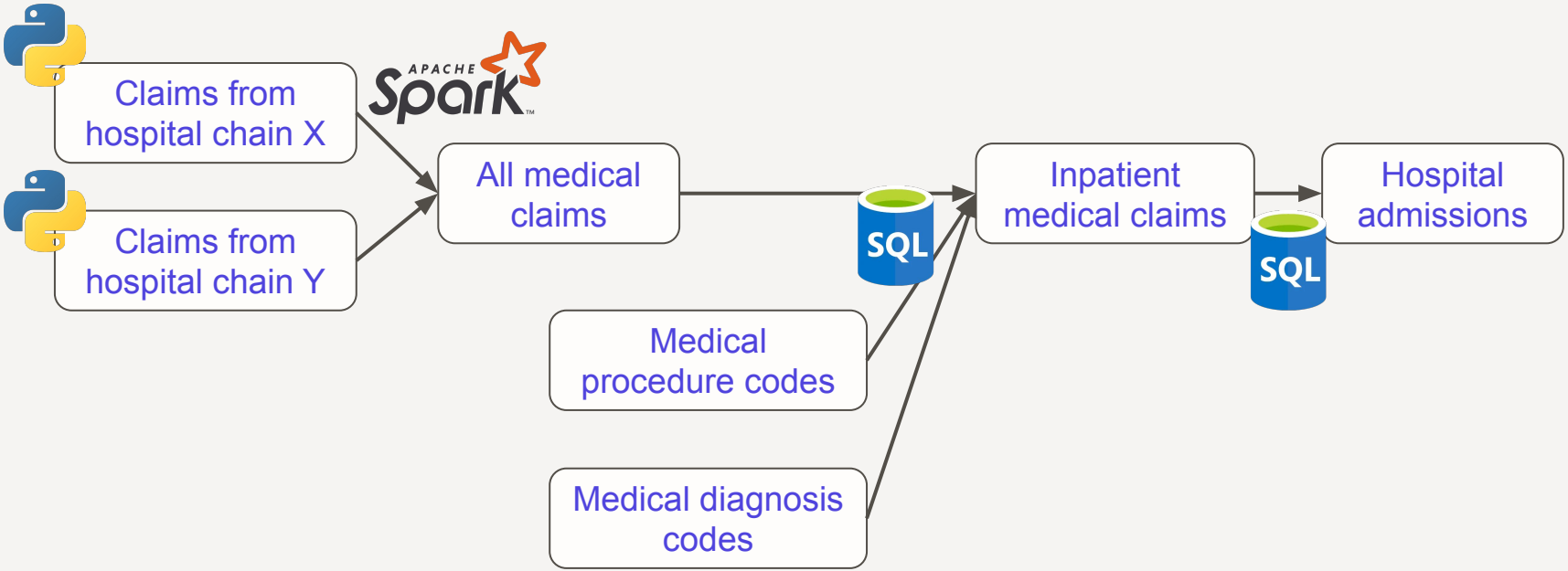


Big complexity

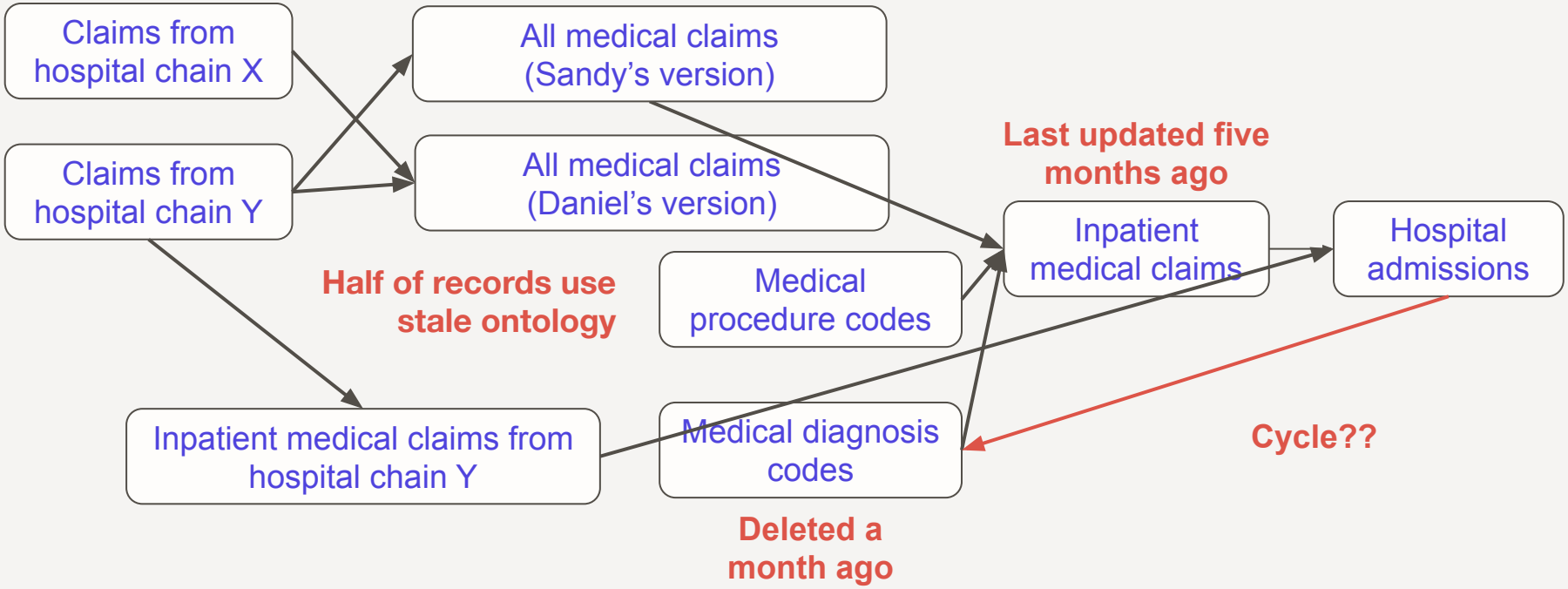
- 100s to 1,000s of tables, ML models, & datasets
- 10,000s of lines of code
- Multiple compute frameworks & storage systems



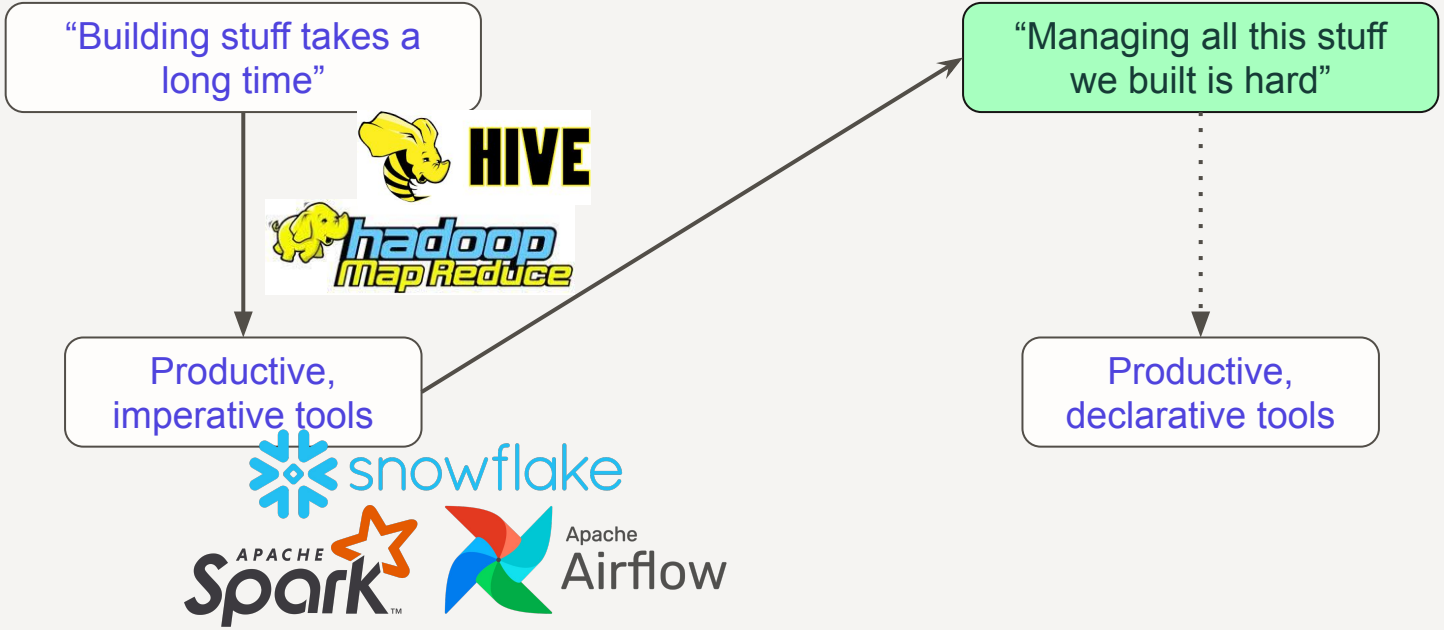
Example: how many hospital admissions?



Complexity » Chaos



The data trajectory

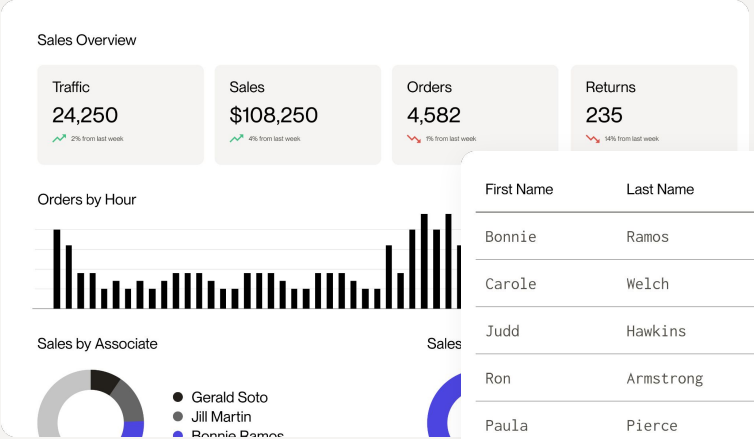


Declarative data management

What’s the declarative entity?

Frontend	→	`UI component`
Dev Ops	→	`Resource`
Data	→	`???`

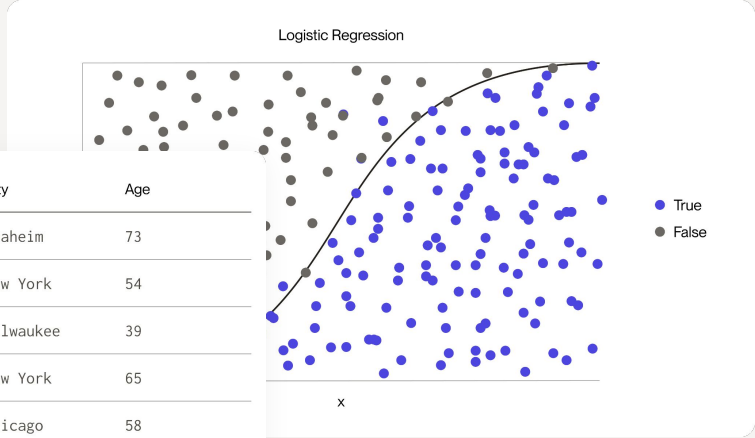
The asset!



Report

First Name	Last Name	Address	City	Age
Bonnie	Ramos	5530 E Pecan St	Anaheim	73
Carole	Welch	1931 Mcclellan Rd	New York	54
Judd	Hawkins	9532 Preston Rd	Milwaukee	39
Ron	Armstrong	6606 Cherry St	New York	65
Paula	Pierce	7674 Bruce St	Chicago	58
Bradley	Wallace	4896 W 6th St	Austin	61
Cat	Fowler	63 Pecan Acres Ln	New York	32
Courtney	Austin	874 Oak Lawn Ave	Chicago	28

Table



ML Model

Introducing the software-defined asset

Asset Definitions

Asset 1

Asset 3

Asset Materializations

Asset 1

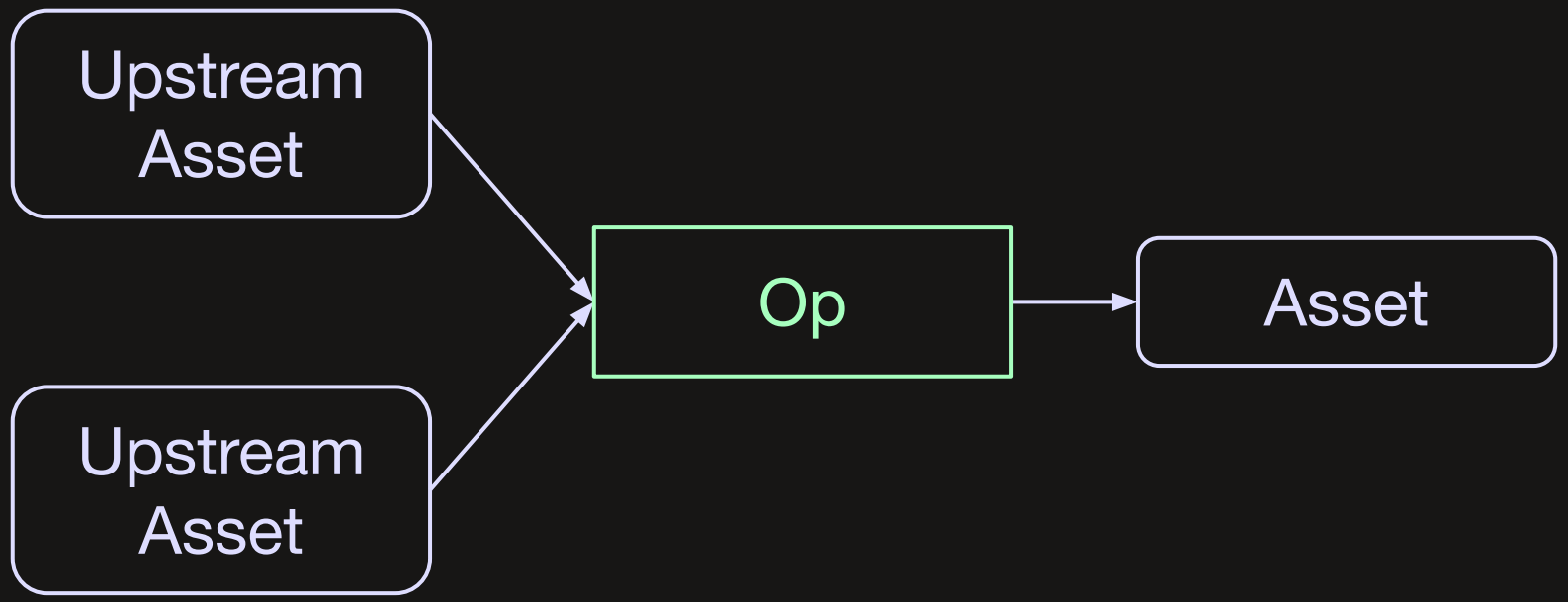
Stale

Asset 2

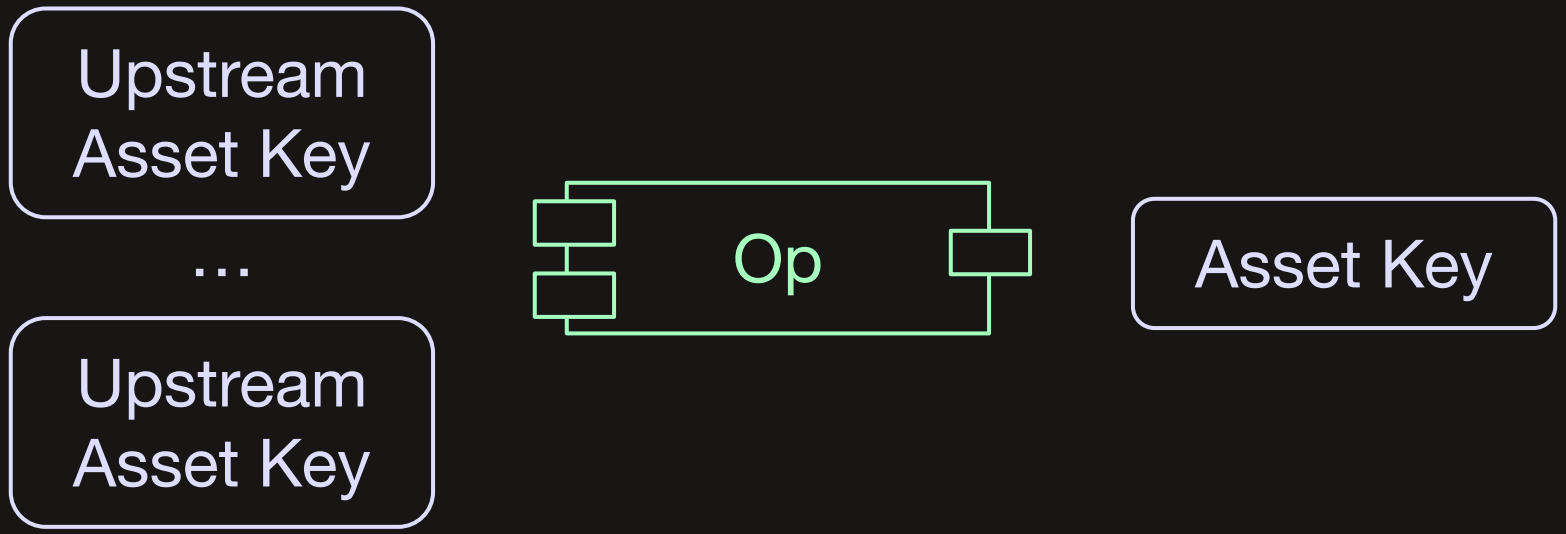
dagster

Software-defined asset

Specifies an asset that you intend to exist, and how to compute it.


Upstream Asset
Upstream Asset
Op
Asset

Software-Defined Asset



Defining an asset in Python

asset key

upstream asset key

```python
from dagster import asset


@asset
def activity_forecast(activity_daily_stats: DataFrame) -> DataFrame:
    """Forecast of activity for the next 30 days"""
    start_date = activity_daily_stats.date.max()
    future_dates = date_range(start=start_date, end=start_date + DateOffset(days=30))
    predicted_data = 0.5 * np.exp(7 * (future_dates.astype(np.int64) / 10**18 - 1.6095))
    return DataFrame({"date": future_dates, "num_comments": predicted_data})
```

compute function

Python isn’t the only way to define an asset…

Defining an asset in SQL (with dbt)

asset key

upstream asset keys

activity_daily_stats.sql

```
select *
from {{ ref('comment_daily_stats') }}
full outer join {{ ref('story_daily_stats') }}
using (date)
```

compute function

activity_daily_stats.sql

```sql
select *
from {{ ref('comment_daily_stats') }}
full outer join {{ ref('story_daily_stats') }}
using (date)
```

```python
@asset
def activity_forecast(activity_daily_stats: DataFrame) -> DataFrame:
    """Forecast of activity for the next 30 days"""
    start_date = activity_daily_stats.date.max()
    future_dates = date_range(start=start_date, end=start_date + DateOffset(days=30))
    predicted_data = 0.5 * np.exp(7 * (future_dates.astype(np.int64) / 10**18 - 1.6095))
    return DataFrame({"date": future_dates, "num_comments": predicted_data})
```

```python
from dagster_dbt.asset_defs import load_assets_from_dbt_manifest
```

```python
dbt_assets = load_assets_from_dbt_manifest(
    json.load(open(os.path.join(DBT_PROJECT_DIR, "target", "manifest.json"))),
    io_manager_key="warehouse_io_manager",
)
```

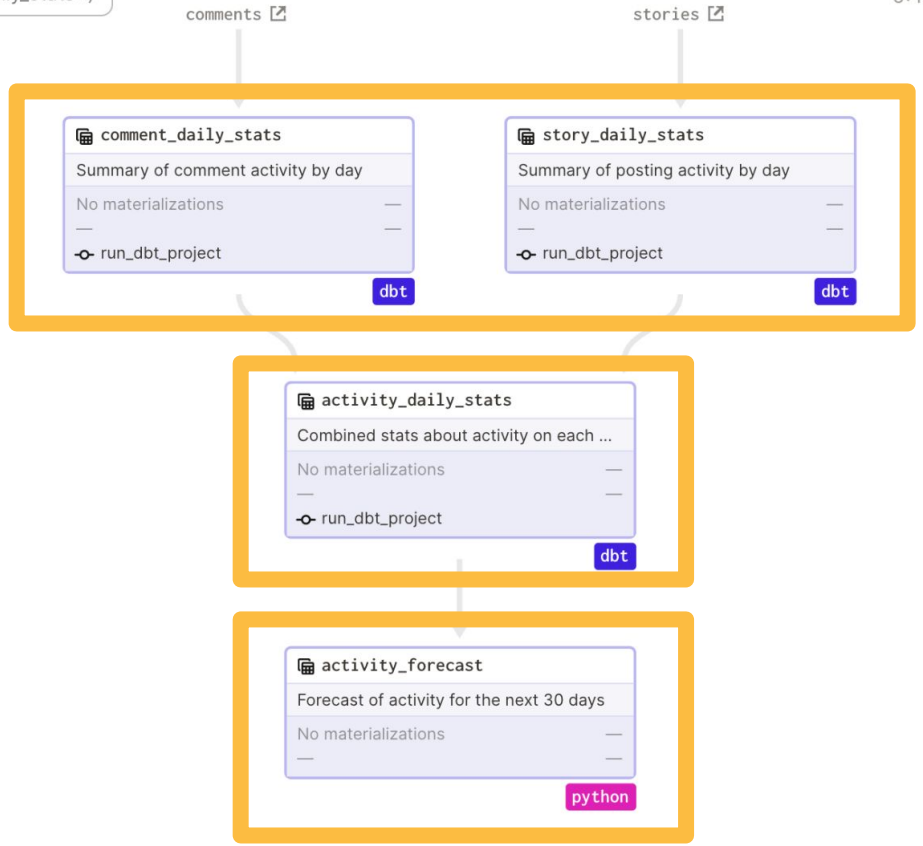
```python
@repository
def activity_analytics():
    return [
        AssetGroup.from_package_name(
            "hacker_news_assets.assets.activity_analytics", resource_defs=RESOURCES_LOCAL
        )
    ]
```

activity_analytics_assets Job in activity_analytics@hacker_news_assets

Overview Launchpad Runs

Type an asset subset... (ex: activity_daily_stats+)

0:13 Materialize All



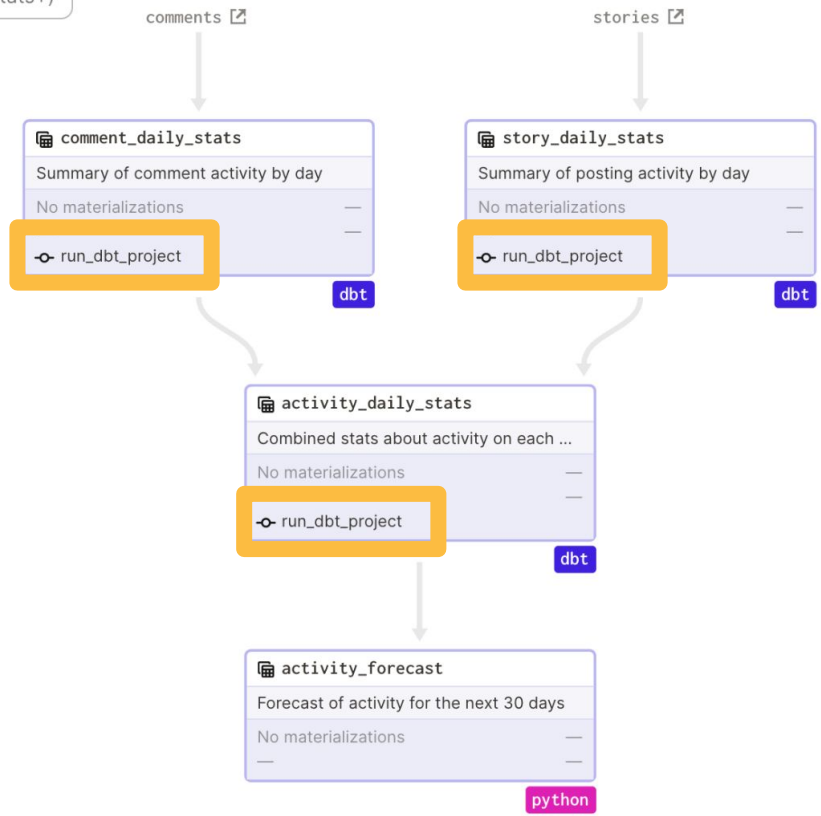
View as Asset Graph

activity_analytics_assets Job in activity_analytics@hacker_news_assets

Overview Launchpad Runs

Type an asset subset... (ex: activity_daily_stats+)

0:13 Materialize All



View as Asset Graph

Decentralized dependencies → more scalable graph

activity_analytics_assets

Job in activity_analytics@hacker_news_assets

Overview Launchpad Runs

Type an asset subset... (ex: activity_daily_stats+)

0:13

Materialize All

View as Asset Graph

activity_analytics_assets

Job in activity_analytics@hacker_news_assets

Overview Launchpad Runs

Type an asset subset... (ex: activity_daily_stats+)

0:13

Materialize All

comments

stories

comment_daily_stats

Summary of comment activity by day

No materializations

run_dbt_project

dbt

story_daily_stats

Summary of posting activity by day

No materializations

run_dbt_project

dbt

activity_daily_stats

Combined stats about activity on each ...

No materializations

run_dbt_project

dbt

activity_forecast

Forecast of activity for the next 30 days

No materializations

python

View as Asset Graph

b46f63d2 Success Run of activity_analytics_assets @ d1234f7b

View tags and config | Debug file

Hide not started steps | Re-execute All (*)

5s 10s 15s 20s 25s 30s 35s 40s

dbt_project

Type a step subset (ex: ++activity_forecast) | Hide unselected steps

Not executed (0)

No steps are waiting to execute

Executing (0)

No steps are executing

Errored (0)

Filter... | debug | info | warning | error | critical | event | Copy URL

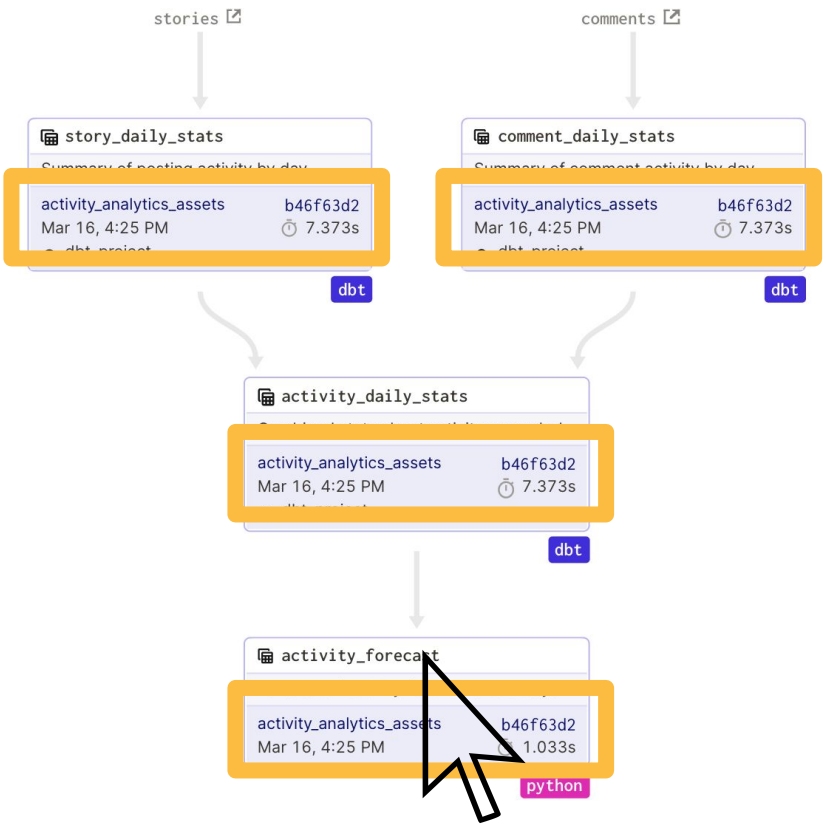
OP	EVENT TYPE	INFO	TIMESTAMP
activity_forecast	LOGS_CAPTURED	Started capturing logs for step: activity_forecast. captured_logs: View stdout / stderr	16:25:16.968
activity_forecast	STEP_START	Started execution of step "activity_forecast".	16:25:17.086
activity_forecast	LOADED_INPUT	Loaded input "activity_daily_stats" using input manager "warehouse_io_manager", from output "activity_daily_stats" of step "dbt_project"	16:25:17.208
activity_forecast	STEP_INPUT	Got input "activity_daily_stats" of type "DataFrame". (Type check passed).	16:25:17.261
activity_forecast	STEP_OUTPUT	Yielded output "result" of type "DataFrame". (Type check passed).	16:25:17.788
activity_forecast	ASSET_MATERIALIZAT…	Materialized value activity_forecast. asset_key: activity_forecast [View Asset]	16:25:17.889
activity_forecast	HANDLED_OUTPUT	Handled output "result" using IO manager "io_manager"	16:25:18.067
activity_forecast	STEP_SUCCESS	Finished execution of step "activity_forecast" in 807ms.	16:25:18.119
-	ENGINE_EVENT	Multiprocess executor: parent process exiting after 38.84s (pid: 55765) pid: 55765	16:25:18.833
-	RUN_SUCCESS	Finished execution of run for "activity_analytics_assets".	16:25:18.907
-	ENGINE_EVENT	Process for run exited (pid: 55765).	16:25:19.031

activity_analytics_assets Job in activity_analytics@hacker_news_assets

Overview Launchpad Runs

Type an asset subset... (ex: story_daily_stats+)

0:01 Rematerialize All



View as Asset Graph

activity_forecast Asset in activity_analytics@hacker_news_assets

0:11 Rematerialize

Activity Definition

Asset Events

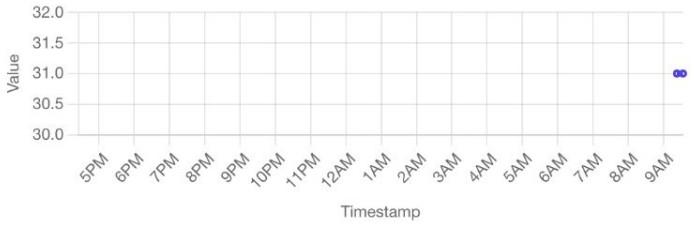
Timestamp	Job / Pipeline	Run
Mar 17, 9:32 AM Materialization		0e4f7d91
Mar 17, 9:22 AM Materialization		3f7a7370

Row count	31
Path	/tmp/activity_forecast.pq
Sample	

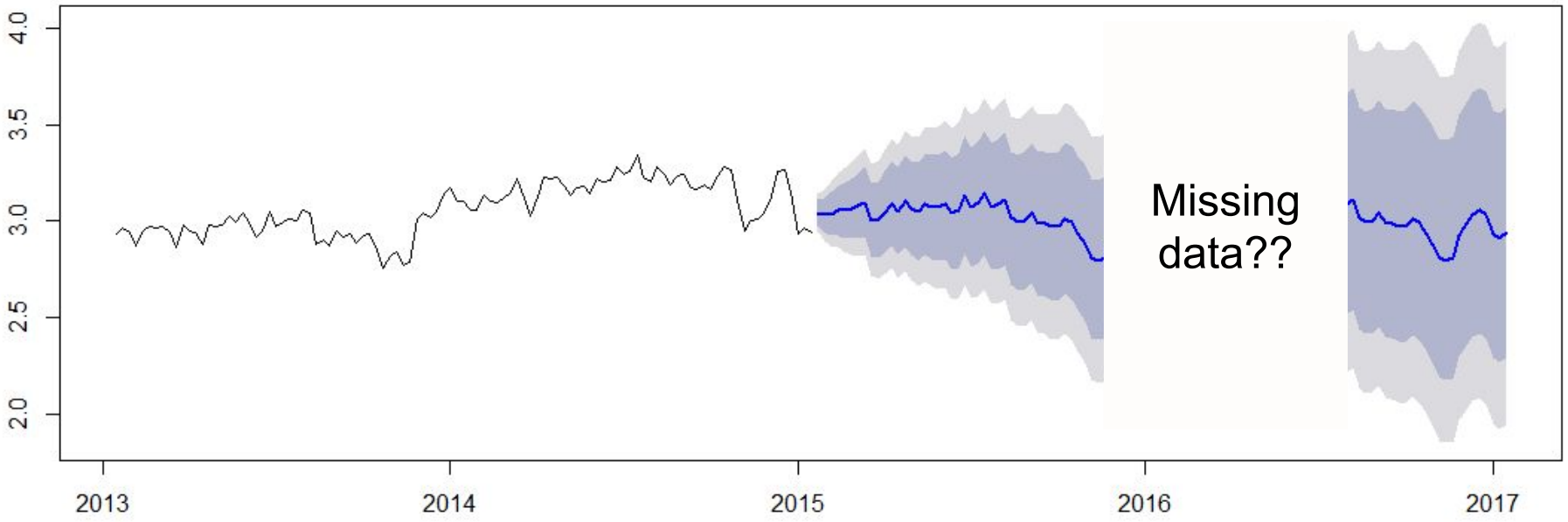
	date	num_comments
0	2022-03-16 00:00:00	0.65186
1	2022-03-17 00:00:00	0.652254
2	2022-03-18 00:00:00	0.652649
3	2022-03-19 00:00:00	0.653044
4	2022-03-20 00:00:00	0.653439

Columns	date datetime64[ns] num_comments float64

Row count



```
select * from activity_forecast
```



activity_forecast

Asset in activity_analytics@hacker_news_assets

0:02

Rematerialize

Activity | Definition

activity_forecast

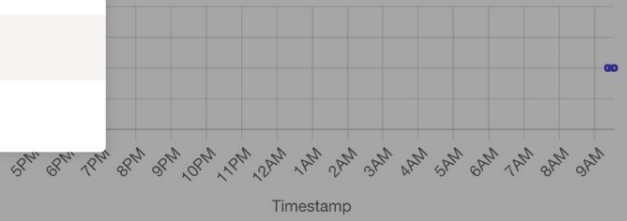
activity_analytics_assets
Job

activity_forecast
Asset

activity_daily_stats
Asset

Asset Events

Timestamp	Job / Pipeline
Mar 17, 9:32 AM Materialization	
Mar 17, 9:22 AM Materialization	



Row count	31
Path	/tmp/activity_forecast.pq

Sample

	date	num_comments
0	2022-03-16 00:00:00	0.65186
1	2022-03-17 00:00:00	0.652254
2	2022-03-18 00:00:00	0.652649
3	2022-03-19 00:00:00	0.653044
4	2022-03-20 00:00:00	0.653439

Columns

date	datetime64[ns]
num_comments	float64

activity_forecast Asset in activity_analytics@hacker_news_assets

0:11 Rematerialize

Activity Definition

Asset Events

Timestamp	Job / Pipeline	Run
Mar 17, 9:32 AM Materialization		0e4f7d91
Mar 17, 9:22 AM Materialization		3f7a7370

Row count	31
Path	/tmp/activity_forecast.pq

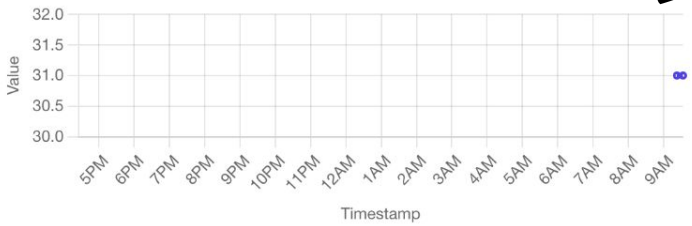
Sample

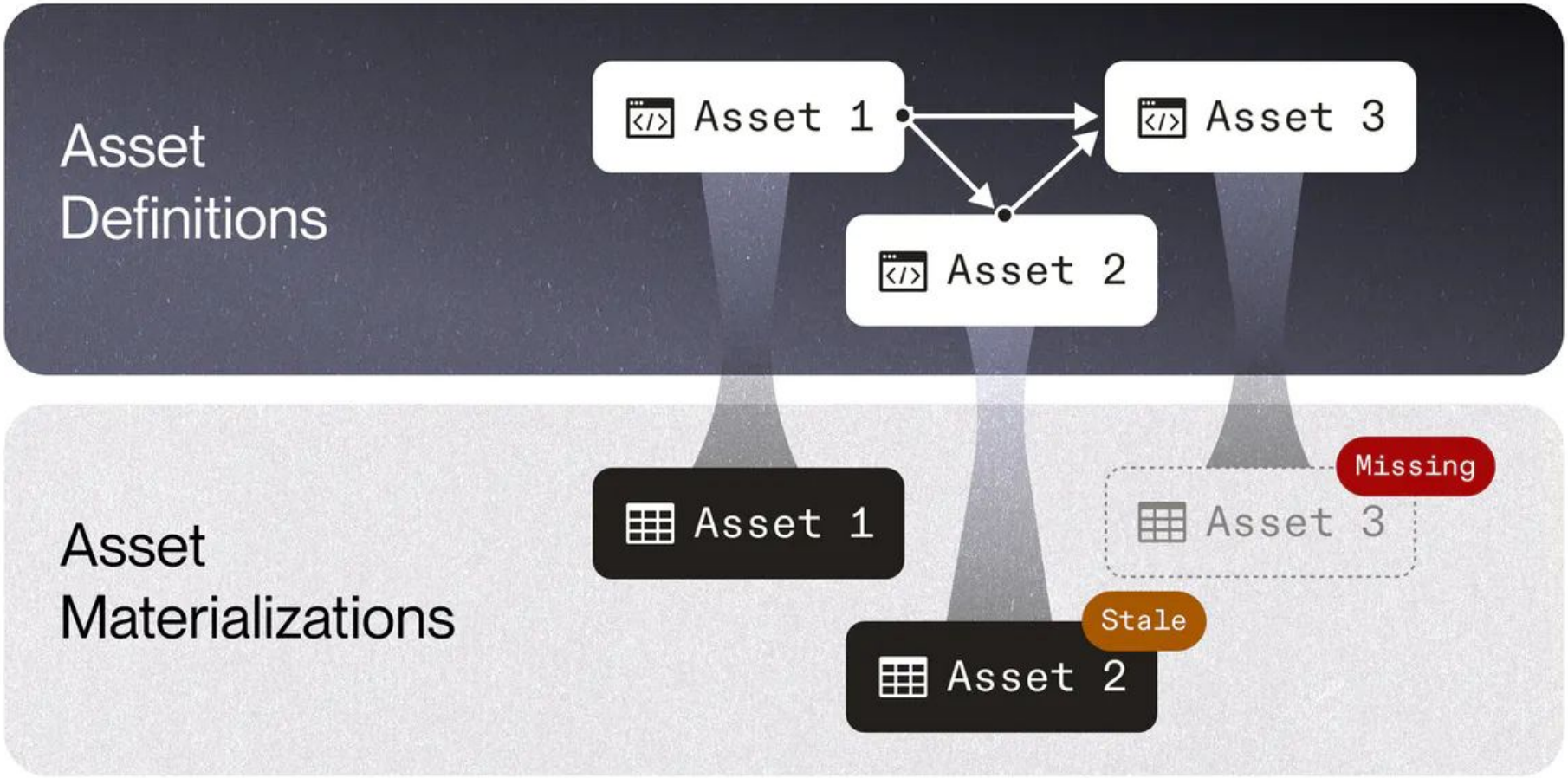
	date	num_comments
0	2022-03-16 00:00:00	0.65186
1	2022-03-17 00:00:00	0.652254
2	2022-03-18 00:00:00	0.652649
3	2022-03-19 00:00:00	0.653044
4	2022-03-20 00:00:00	0.653439

Columns

date	datetime64[ns]
num_comments	float64

Row count




Asset Definitions
Asset 1
Asset 3
Asset 2
Asset Materializations
Asset 1
Missing
Asset 3
Stale
Asset 2

Assets are not static

Orchestration

What does a traditional orchestrator do?

- Invokes computations at the right time
- Models the dependencies between computations
- Tracks what computations ran

Orchestrators are experts on

- When stuff happens
- When stuff is going to happen
- What it takes to make something happen

Critical questions about our assets

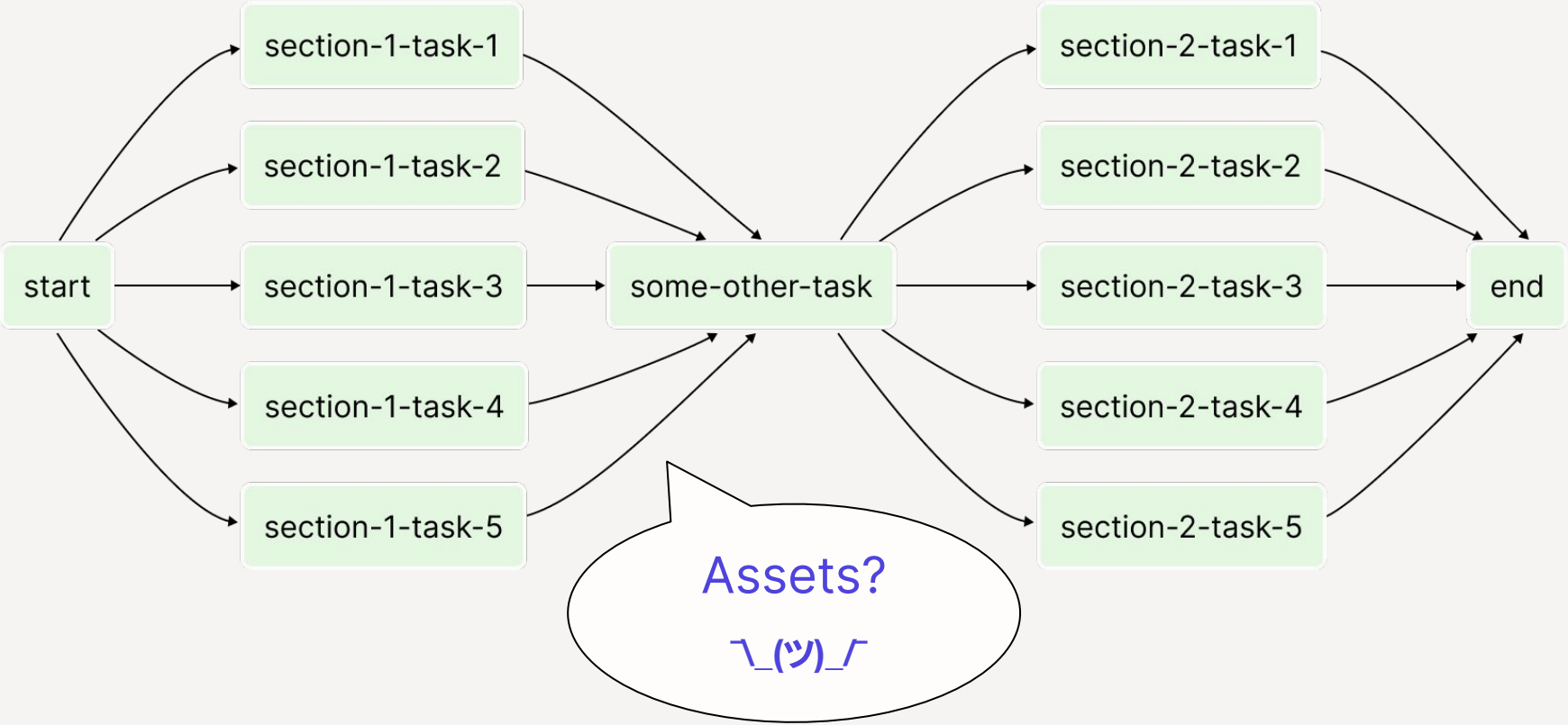
Is this asset up-to-date?

What do I need to run to refresh this asset?

When will this asset be updated next?

What code and data were used to generate this asset?

After pushing a change, what assets need to be updated?


start
section-1-task-1
section-1-task-2
section-1-task-3
section-1-task-4
section-1-task-5
some-other-task
section-2-task-1
section-2-task-2
section-2-task-3
section-2-task-4
section-2-task-5
end
Assets?
¯_(ツ)_/¯

Orchestration + Assets = ?

Asset orchestrator:

manages change in assets

Triggered asset jobs

Re-materialize your assets on a schedule, or via a sensor

Reconciliation

Update your assets when they don’t match their definitions

```
ScheduleDefinition(
    job=assets.build_job(
        "daily_stats_job", selection=["*activity_daily_stats"]
    ),
    cron_schedule="0 0 * * *",
),
```

daily_stats_job

Job in activity_analytics@hacker_news_assets

Schedule: At 12:00 AM

Overview Launchpad Runs

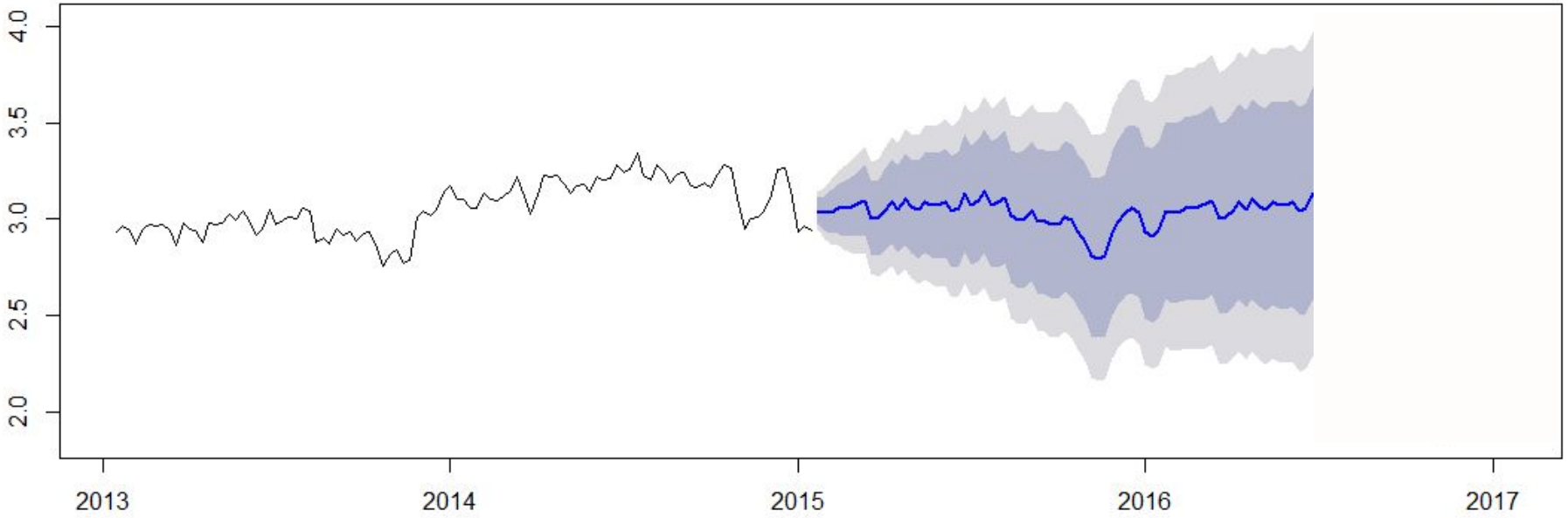
Type an asset subset... (ex: story_daily_stats+*)

0:08

Materialize All

View as Asset Graph

```
select * from activity_forecast
```



activity_daily_stats

Asset in activity_analytics@hacker_news_assets

Schedule: At 12:00 AM

Activity | Definition

Description

daily_stats_job run_dbt_project

Combined stats about activity on each day

Raw SQL:

```
select *
from {{ ref('comment_daily_stats') }}
full outer join {{ ref('story_daily_stats') }}
using (date)
```

Upstream Assets (2)

View upstream graph

story_daily_stats
No materializations
run_dbt_project

comment_daily_stats
No materializations
run_dbt_project

Downstream Assets (1)

View downstream graph

activity_forecast
No materializations

Critical questions about our assets

Is this asset up-to-date?

What do I need to run to refresh this asset?

When will this asset be updated next?

What code and data were used to generate this asset?

After pushing a change, what assets need to be updated?

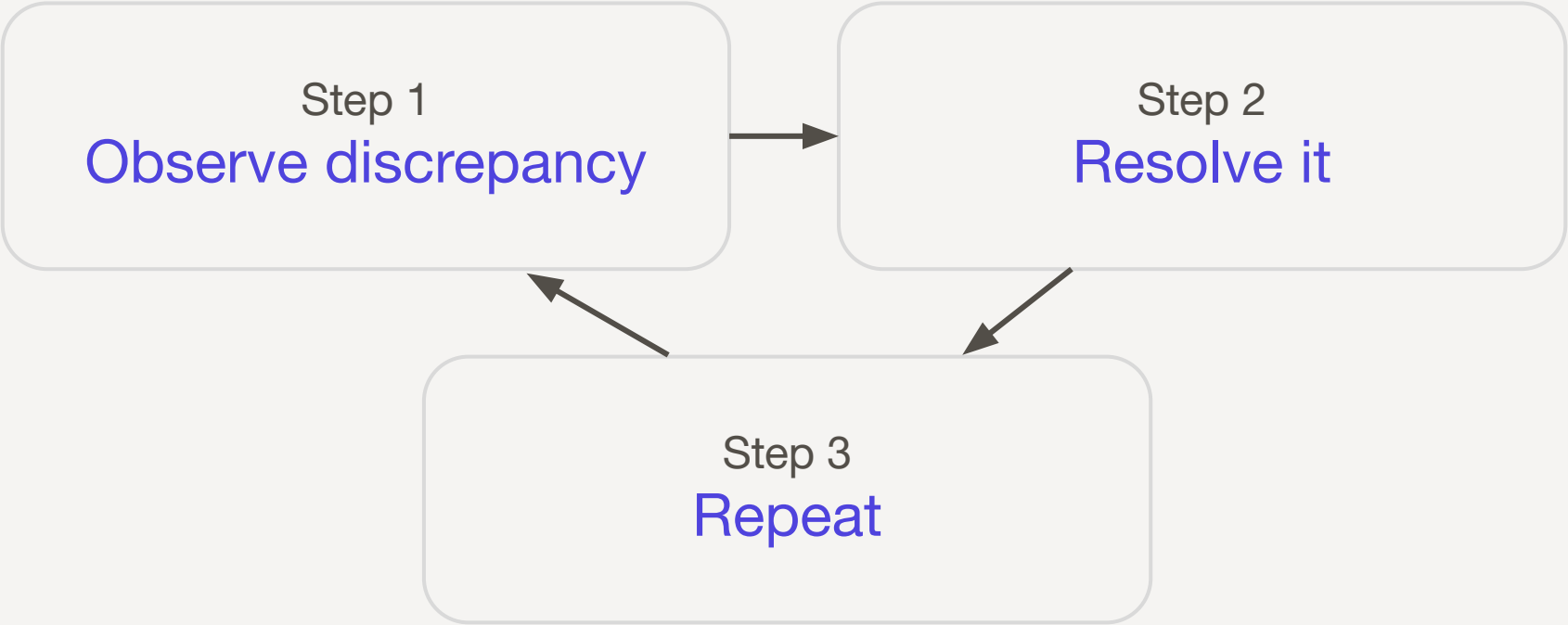
Scheduled asset jobs

Re-materialize your assets on a schedule

Reconciliation

Update your assets when they don’t match their definitions

Reconciliation: how it works



Discrepancy: definition w/o materialization

Discrepancy: partition definition w/o materialization

items Asset in core@hacker_news_assets

Activity **Definition**

Description

Items from the Hacker News API: each is a story or a comment on a story.

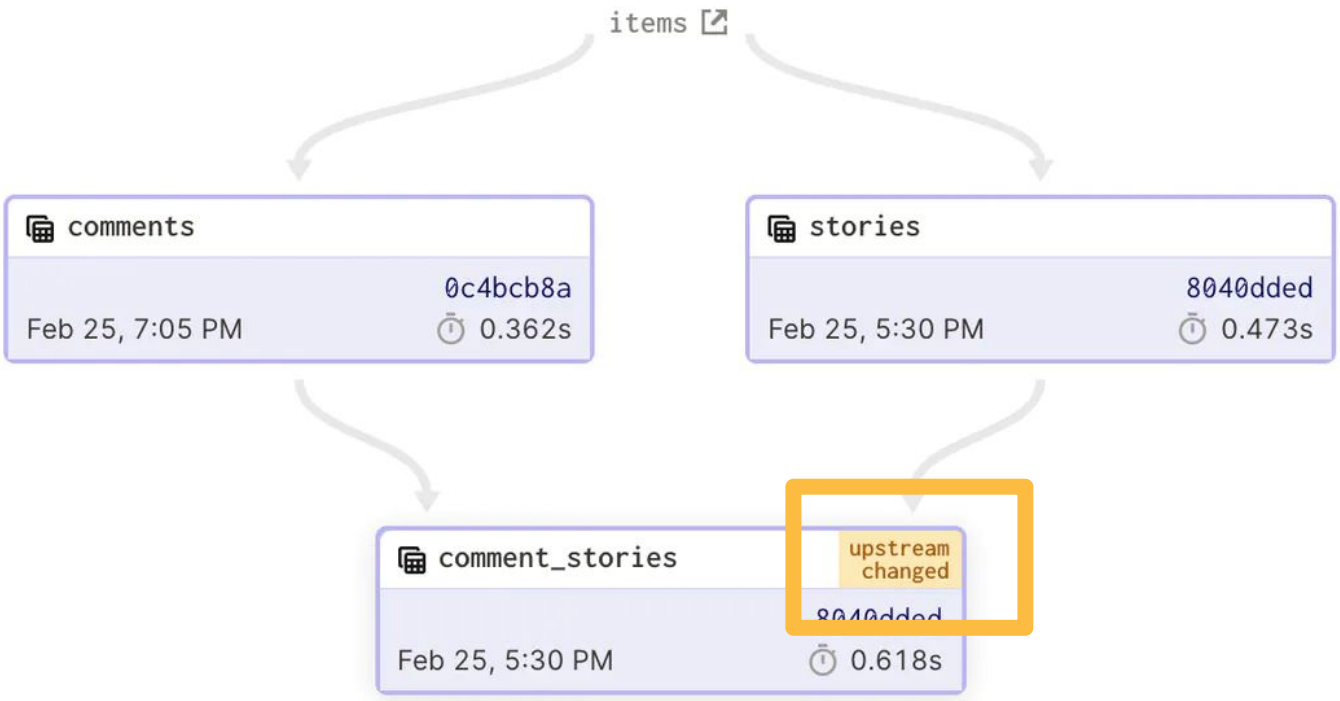
Partitions

Hourly, starting 2022-03-18-12:00 UTC.

3/10 Partitions

2022-03-18-12:00 2022-03-18-21:00

Discrepancy: upstream changed



Discrepancy: columns

Definition

Type

LoginsSchema (table)

event_id str non-nullable unique
The ID of the event.

session_id str unique
The ID of the session that includes the login.

user_id str non-nullable
The ID for the user who logged in.

timestamp datetime64[ns] non-nullable
The timestamp of the login (UTC).

attempts int64 non-nullable
The number of login attempt before success.

Latest Materialization

Asset Events

Timestamp

Feb 28, 9:08 AM
Materialization

schema

event_id str non-nullable unique
The ID of the event.

session_id str unique
The ID of the session that includes the login

user_id str non-nullable
The ID for the user who logged in.

timestamp datetime64[ns] non-nullable
The timestamp of the login (UTC).

The fruits of reconciliation

- Move bookkeeping about what’s stale out of your head
- Avoid unnecessary computation
- Understand why computations occurred

Software-defined assets & the modern data stack

A set of tools and practices that have dramatically simplified common patterns for working with data.

Define derived tables

Define source tables

Pre-modern world

Execution	Ingest	SQL transforms	Python transforms	ML
Orchestration	Apache Airflow			

Modern world

Execution

Orchestration

Fivetran, meltano, Airbyte	dbt		Continual

Sync external tables @ midnight

Rebuild derived tables @ 1 am


dreamstime

We’re forced to choose…

Unified b

(Pre-moc

ut siloed

Data Stack)

Unified and Declarative

dagster

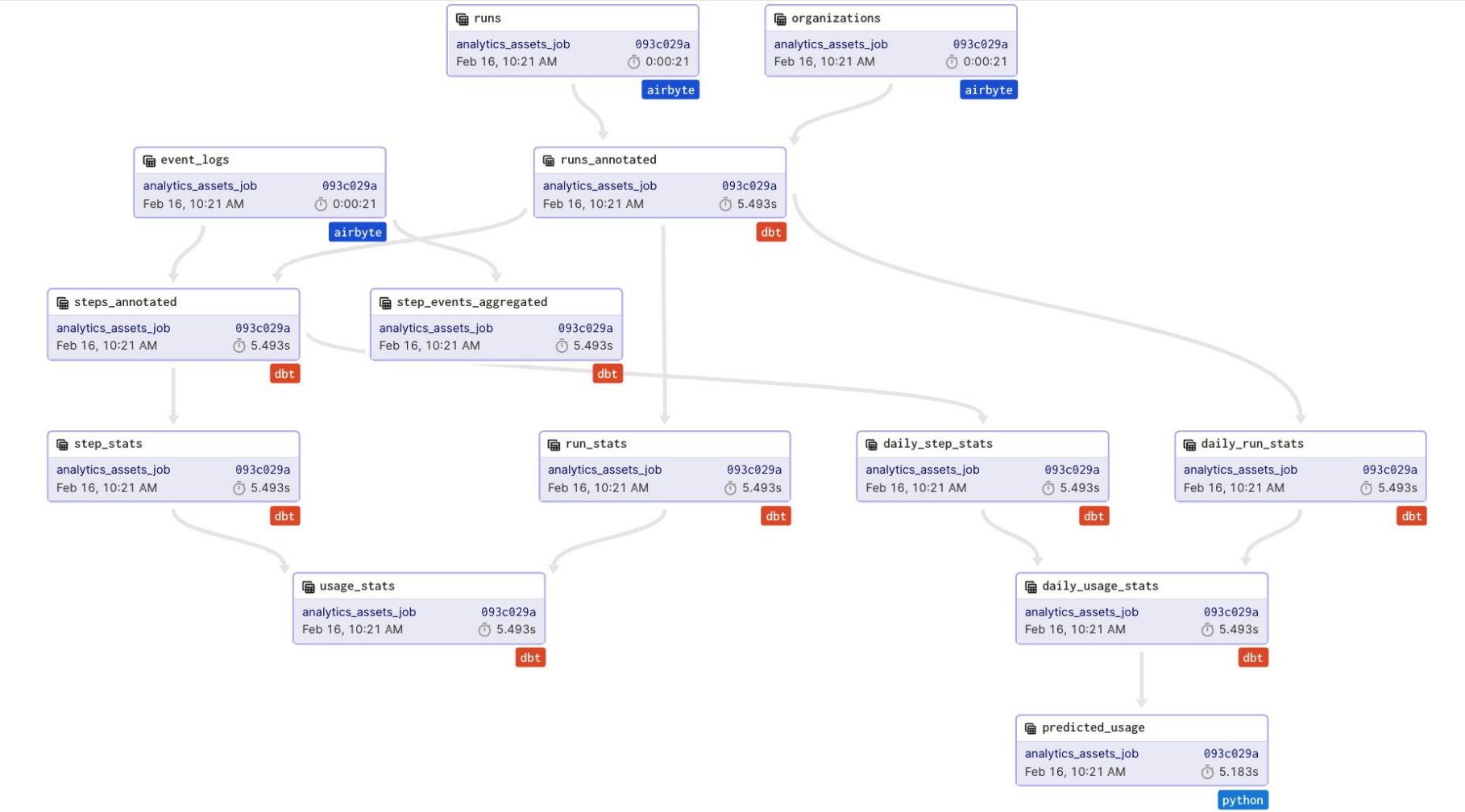
Airbyte

```
airbyte_assets = build_airbyte_assets(
    connection_id="my-airbyte-connection-id",
    destination_tables=["table1", "table2"],
)
```

dbt

```
dbt_assets = load_assets_from_dbt_project(
   project_dir="my/dbt/project/dir/",
)
```

```
@asset
def predicted_usage(daily_usage_stats: DataFrame) -> DataFrame:
    ...
```


runs
analytics_assets_job 093c029a
Feb 16, 10:21 AM 0:00:21
airbyte
organizations
analytics_assets_job 093c029a
Feb 16, 10:21 AM 0:00:21
airbyte
event_logs
analytics_assets_job 093c029a
Feb 16, 10:21 AM 0:00:21
airbyte
runs_annotated
analytics_assets_job 093c029a
Feb 16, 10:21 AM 5.493s
dbt
steps_annotated
analytics_assets_job 093c029a
Feb 16, 10:21 AM 5.493s
dbt
step_events_aggregated
analytics_assets_job 093c029a
Feb 16, 10:21 AM 5.493s
dbt
step_stats
analytics_assets_job 093c029a
Feb 16, 10:21 AM 5.493s
dbt
run_stats
analytics_assets_job 093c029a
Feb 16, 10:21 AM 5.493s
dbt
daily_step_stats
analytics_assets_job 093c029a
Feb 16, 10:21 AM 5.493s
dbt
daily_run_stats
analytics_assets_job 093c029a
Feb 16, 10:21 AM 5.493s
dbt
usage_stats
analytics_assets_job 093c029a
Feb 16, 10:21 AM 5.493s
dbt
daily_usage_stats
analytics_assets_job 093c029a
Feb 16, 10:21 AM 5.493s
dbt
predicted_usage
analytics_assets_job 093c029a
Feb 16, 10:21 AM 5.183s
python

Boiling it all down

Missing abstraction: the software-defined asset

Declarative approach → less chaos, more trust

Asset orchestrators manage change in assets

Python: first-class citizen of the Modern Data Stack

Unified control plane for the Modern Data Stack

Thank you

Sandy Ryza
@s_ryz

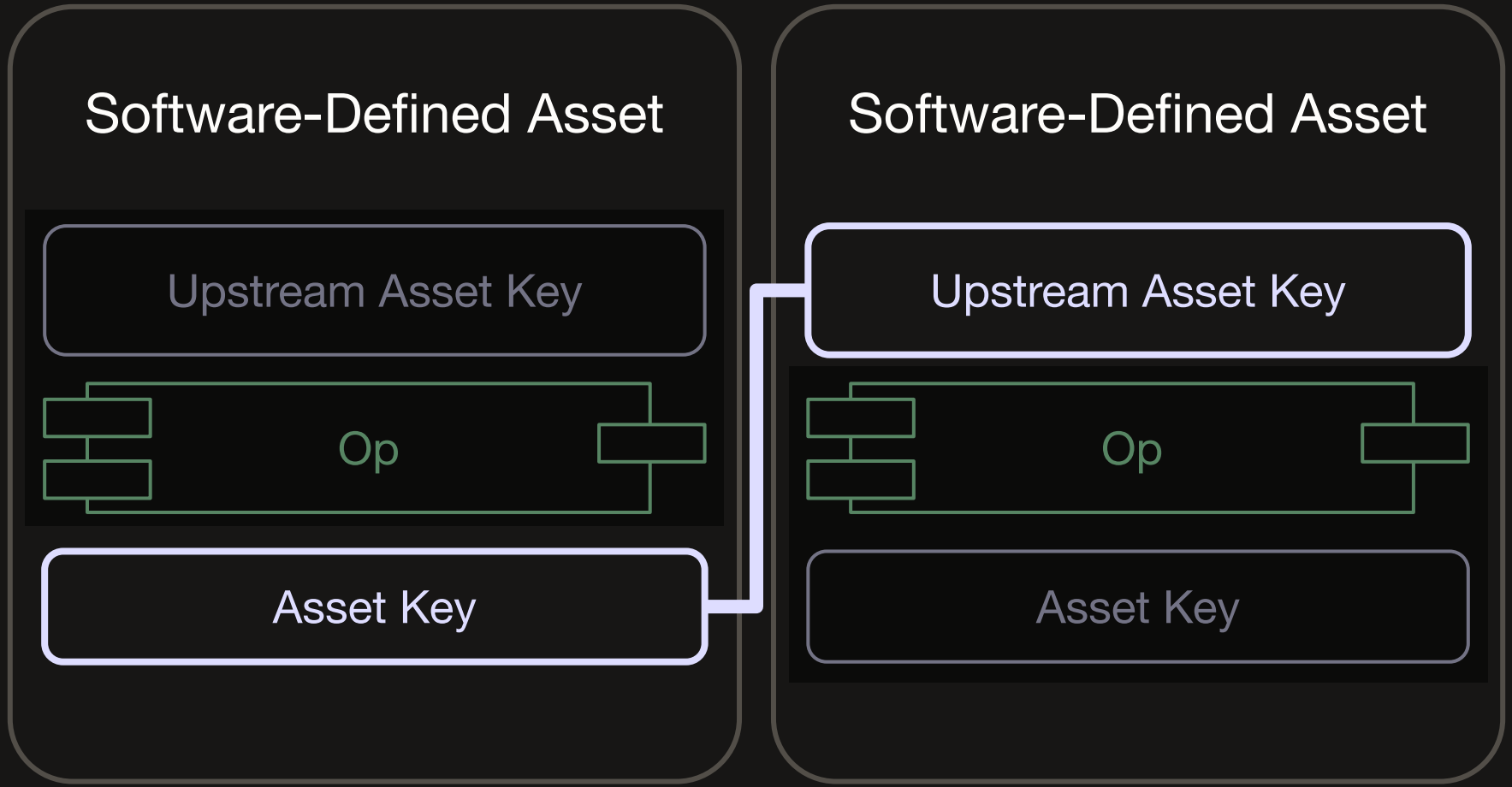
elementl

dagster

dagster

Asset Definitions

Asset Materializations


Software-Defined Asset
Upstream Asset Key
Op
Asset Key
Software-Defined Asset
Upstream Asset Key
Op
Asset Key

The casualties of chaos

Trust

Productivity