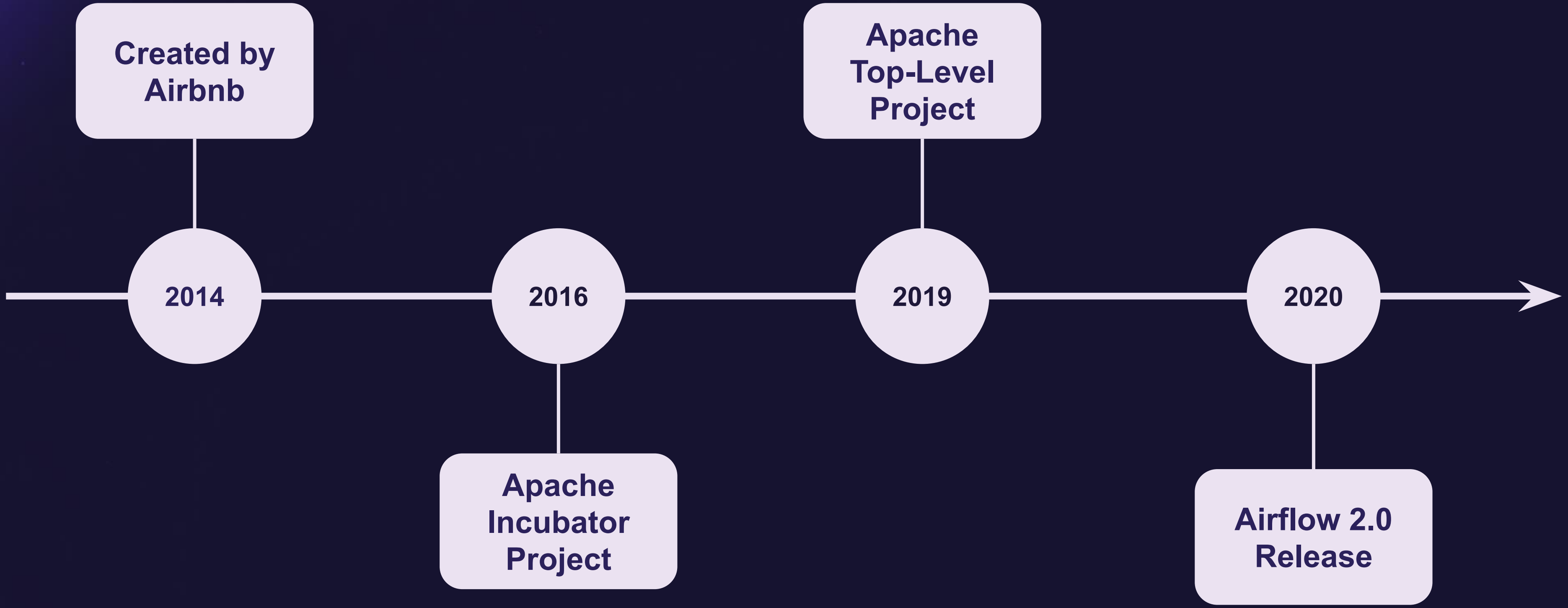
Airflow Summit 2024

Seeing Clearly with Airflow

The Shift to Data-Aware Orchestration

TP Chung Staff Software Engineer
Constance Martineau Senior Product Manager

The Evolution of Apache Airflow



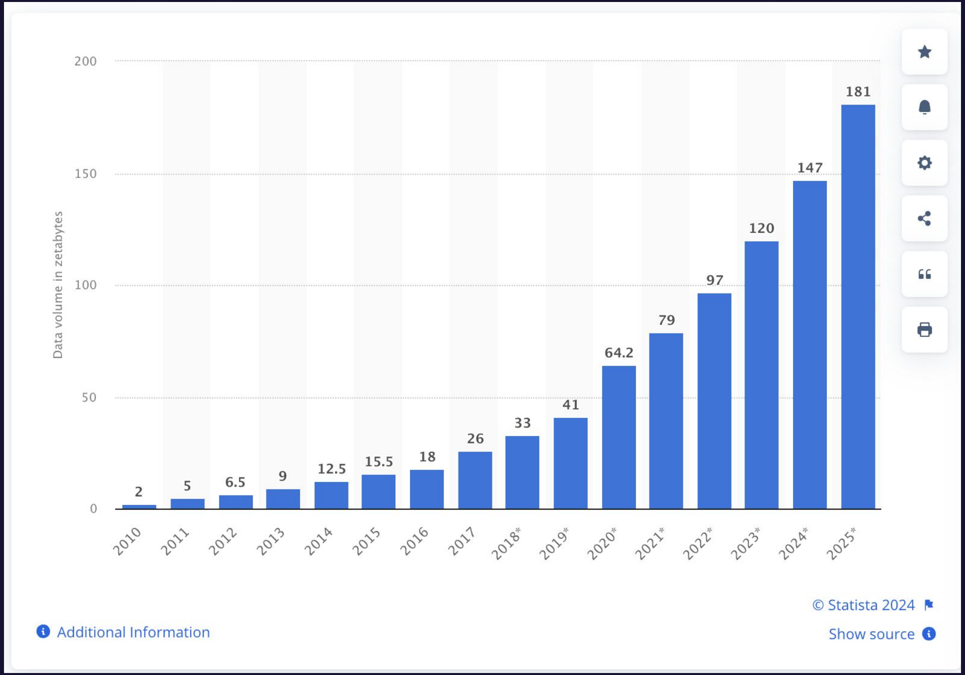
As of September 2024 ...

Apache Airflow

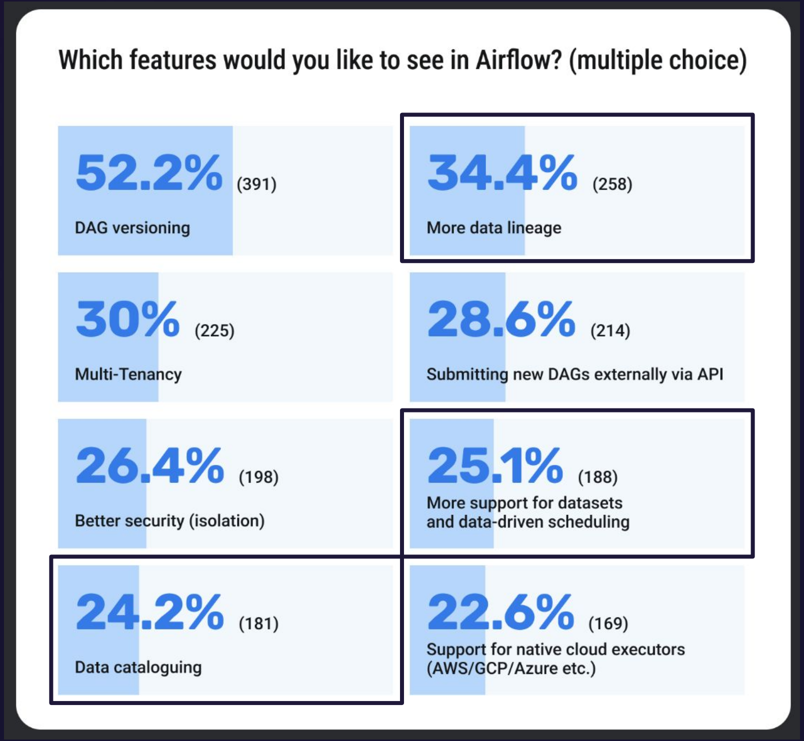
25-30M	Monthly Downloads
3.4K	Contributors
36.2K	Github Stars
45K	Slack Community

Airflow has been successful as a workflow orchestrator by focusing on tasks

Data Growth Worldwide



https://www.statista.com/statistics/871513/worldwide-data-created/



Which features would you like to see in Airflow? (multiple choice)

52.2% (391)
DAG versioning

34.4% (258)
More data lineage

30% (225)
Multi-Tenancy

28.6% (214)
Submitting new DAGs externally via API

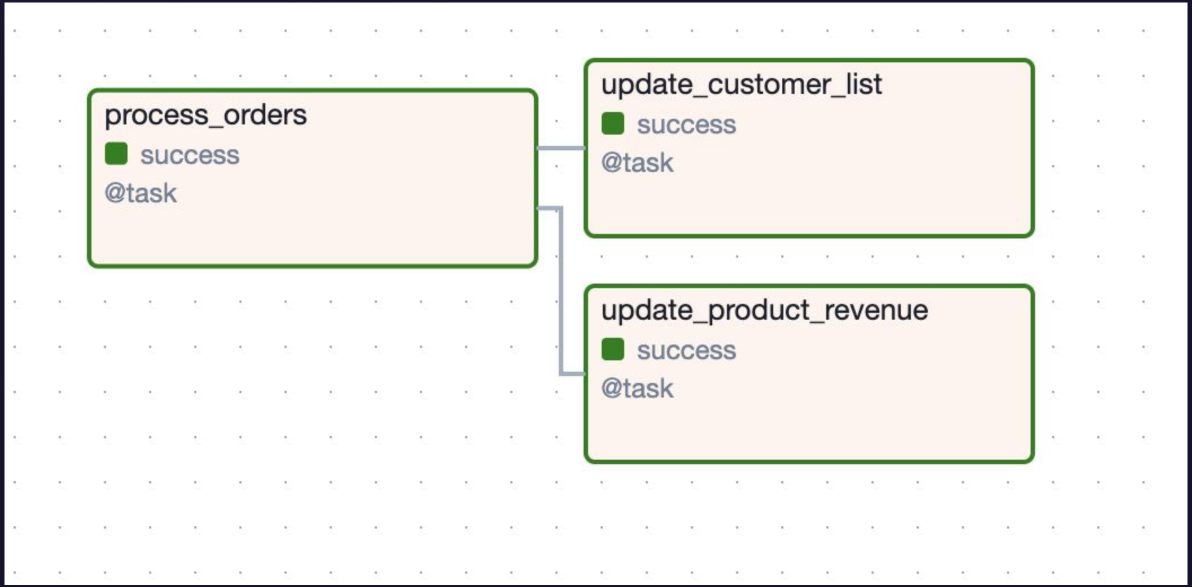
26.4% (198)
Better security (isolation)

25.1% (188)
More support for datasets and data-driven scheduling

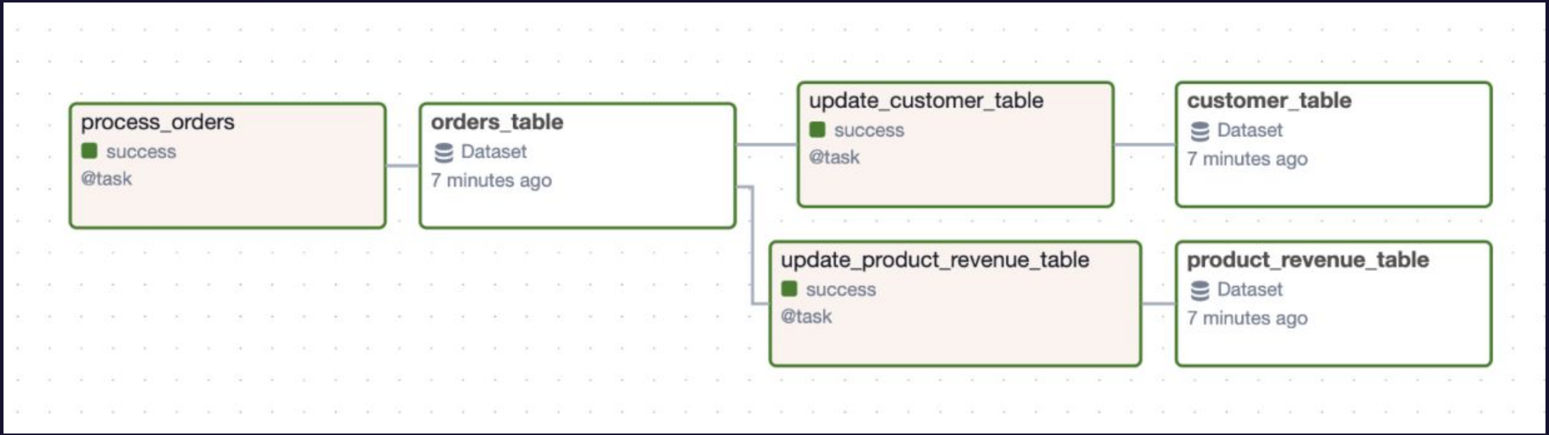
24.2% (181)
Data cataloguing

22.6% (169)
Support for native cloud executors (AWS/GCP/Azure etc.)

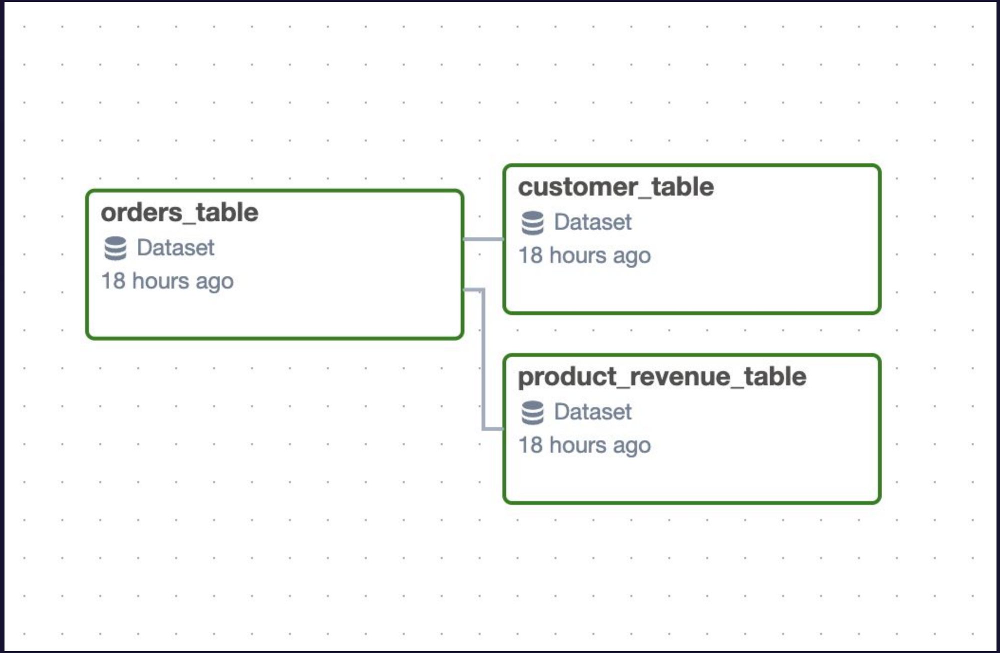
Task-Oriented Pipeline



Task-Oriented Pipelines Using Data-Aware Scheduling



Asset-Oriented Pipelines



By shifting the focus from tasks to data that is produced, Airflow can provide block-level lineage and visibility into data quality

Airflow is Evolving to Meet These Needs

AIP-73: Expanding Data Awareness in Airflow

- AIP-74: Introducing Data Assets
- AIP-75: New Asset-Centric Syntax
- AIP-76: Asset Partitions

Data Assets

Renaming *Dataset*

Dataset → ***Data Asset***

Data assets

Datasets: renamed, enhanced, and more integrated

Dataset renamed

The user-facing class is renamed to better describe the concept and avoid confusion.

Named resource

Assets no longer requires a real URI. They can represent more flexible ideas.

Custom grouping

New “group” specifier allows for categorizing assets logically together.

UI improvements

Asset Views can provide more insights, and be integrated more with other views.

```python
my_model = Asset(name="my_model", group="model")
past_info = Asset(name="past_info", group="dataset")
predictions = Asset(name="prediction", group="result")

@task(outlets=my_model)
def create_model():
    ...

@task(outlets=past_info)
def collect_info():
    ...

@task(inlets=[my_model, past_info], outlets=predictions)
def predict_1():
    # Use the model and available data to make a prediction...
```

New Definition Syntax

```python
@asset(group="model")
def my_model():
    ...

@asset(group="dataset")
def past_info():
    ...

@asset(
    schedule=my_model | past_info,
    group="result",
)
def predictions(my_model, past_info):
    ...
```

ASTRONOMER

Internally...

Three-in-one, but still the same concepts.

It’s a DAG.

More accurately, a *schedulable*; on the same level as a DAG.

It’s a task.

Like Python operator; it executes the same things in the function.

It’s an asset.

Decorated object can track lineage and do event triggering.

```python
@asset(schedule="@monthly", group="model")
def my_model():
    ...

@asset(schedule="@hourly", group="dataset")
def past_info():
    ...

@asset(
    schedule=my_model | past_info,
    group="result",
)
def predictions(my_model, past_info):
    ...
```

ASTRONOMER

```python
...

@asset(
    schedule=my_model | past_info,
    group="result",
)
def predictions(my_model, past_info):
    model_path = ObjectStoragePath(my_model.uri)
    # ... load model from path ...
```

@asset Definitions vs. Good Ol' Tasks

DAG-level definition

An `@asset` is a DAG-level definition. It can't be a part of a DAG. It is a *schedulable* entity.

Noun-based declaration

The `@asset` decorator defines the asset and its materialization. No separate instantiation.

Separate schedule and partition

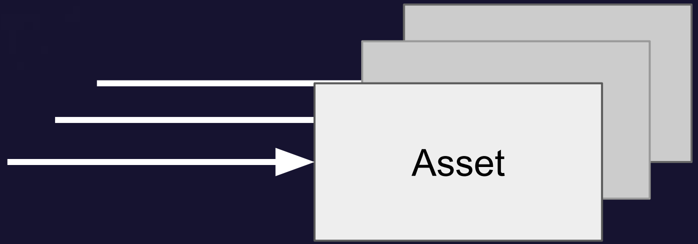
Materialization is scheduled, but logical dates and data intervals don't apply to assets.

Assets don't write themselves!!

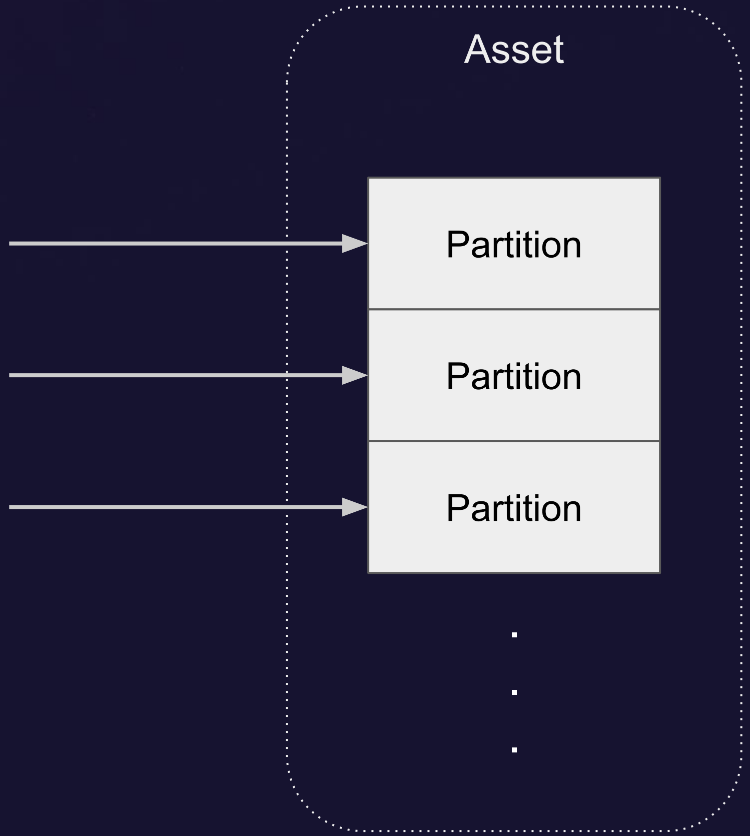
This is the same as a `@task`. The definition only announces the intention. Future work?

Partitions

Simple asset materialization



Partitioned asset materialization



ASTRONOMER

Partitions

How can an asset be partitioned?

Time-based

- ➔ Most common?
- ➔ Data every X
- ➔ Data interval

Data groups

- ➔ Categorization
- ➔ Give it an ID
- ➔ DAG expansion?

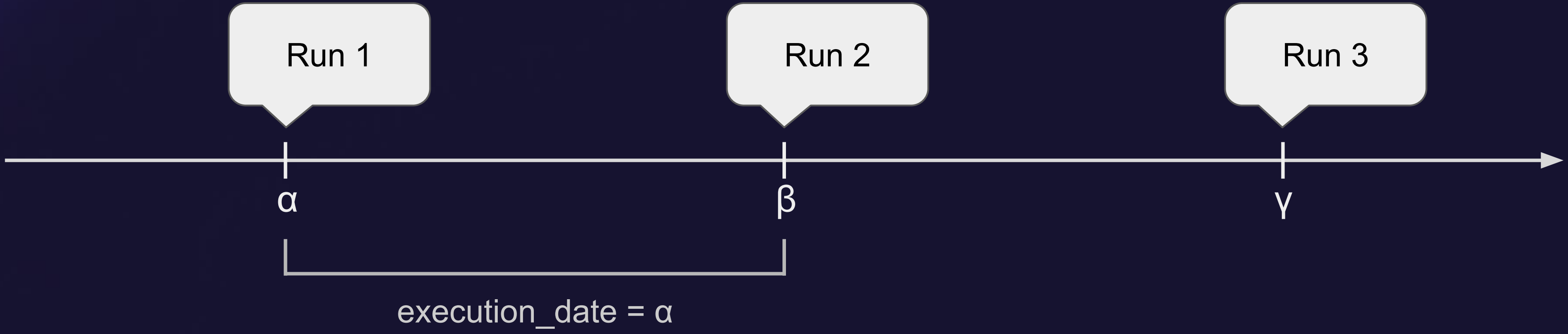
Serial number

- ➔ Time-based-ish
- ➔ A range
- ➔ Dynamically

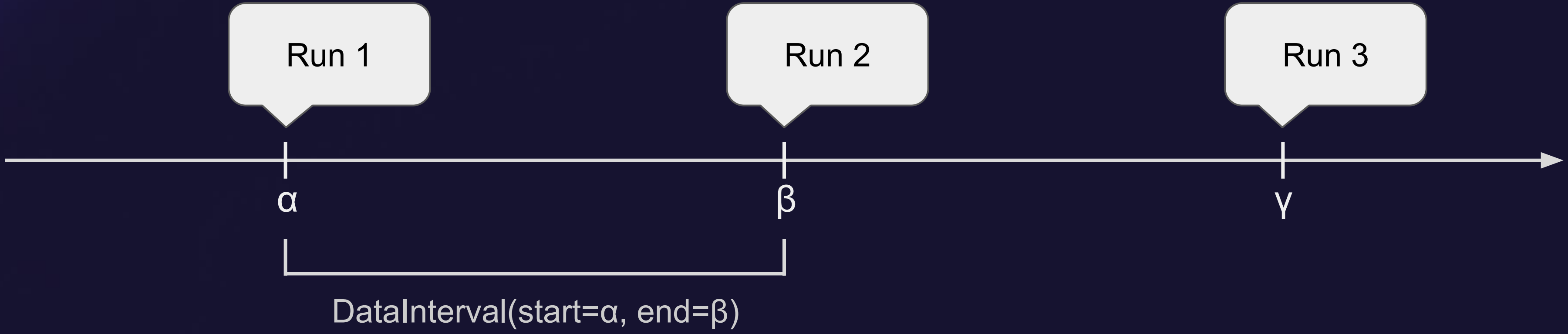
PARTITION IS NOT NEW!

- ➔ All the same principles
- ➔ Reorganized to easier mental model
- ➔ Generalized for more use cases

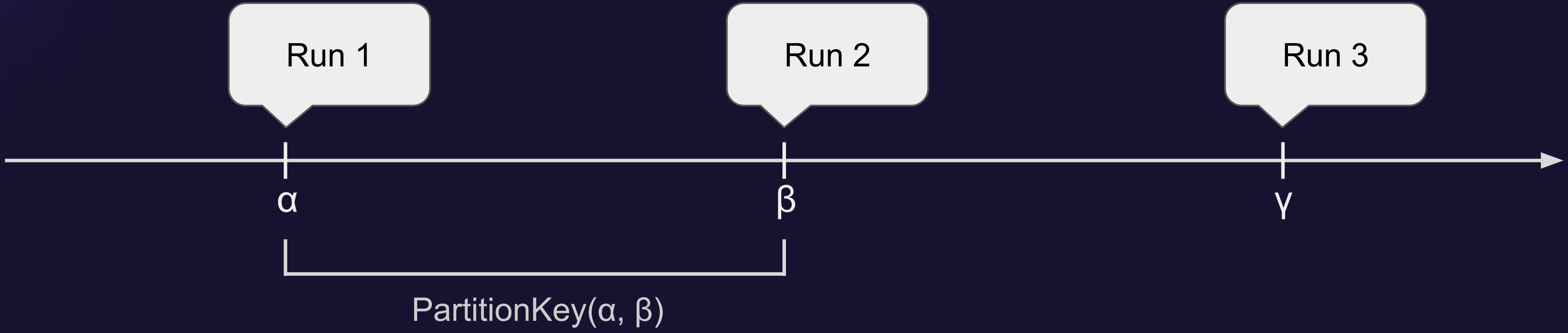
Execution date



Data interval



Partition



Partition

How does the concept improve Airflow?

Simplified

No more *run date* vs *execution date* misunderstandings.

Generalized

Partition key does not need to be a date (or date range).

Independent

Partition does not need to line up with the run schedule.

Takeaways

ASTRONOMER

1. Airflow is evolving to meet the needs of modern data workflows

2. Data Assets and Partitions enable more granular control and transparency

3. The Future of Orchestration is Data-Centric

Thank you!
Any questions?

ASTRONOMER