Is it the end for Apache Airflow?

PyCon LT 2023

Tomas Peluritis/Uncle Data

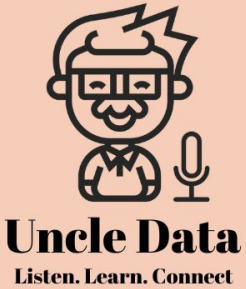
About me

https://www.linkedin.com/in/tomaspeluritis/

https://podcasters.spotify.com/pod/show/duomenu-dede

https://github.com/uncledata

https://uncledata.substack.com



Agenda

- Challenge
- Apache Airflow
- Dagster
- Mage
- Comparisons

Zach Wilson • 1st

Data Eng Creator | ADHD | Advisor@Mage,SylphAI | @EcZachly on TT,IG,T...

2d •

Some unstoppable trends:

- tech layoffs creating a surge in flexible work and solopreneurship with platforms like SylphAI and topmate.io

- data engineering becoming both more software engineering-focused as data products become more mainstream

- simpler data engineering becoming less software focused with low-code solutions like Fivetran

- Airflow becoming the “weakest link” in the data engineering stack and replaced by popular competitors like Mage, Prefect and Dagster

- MLOps getting a lot more attention as more and more companies learn the pains of running a model in production

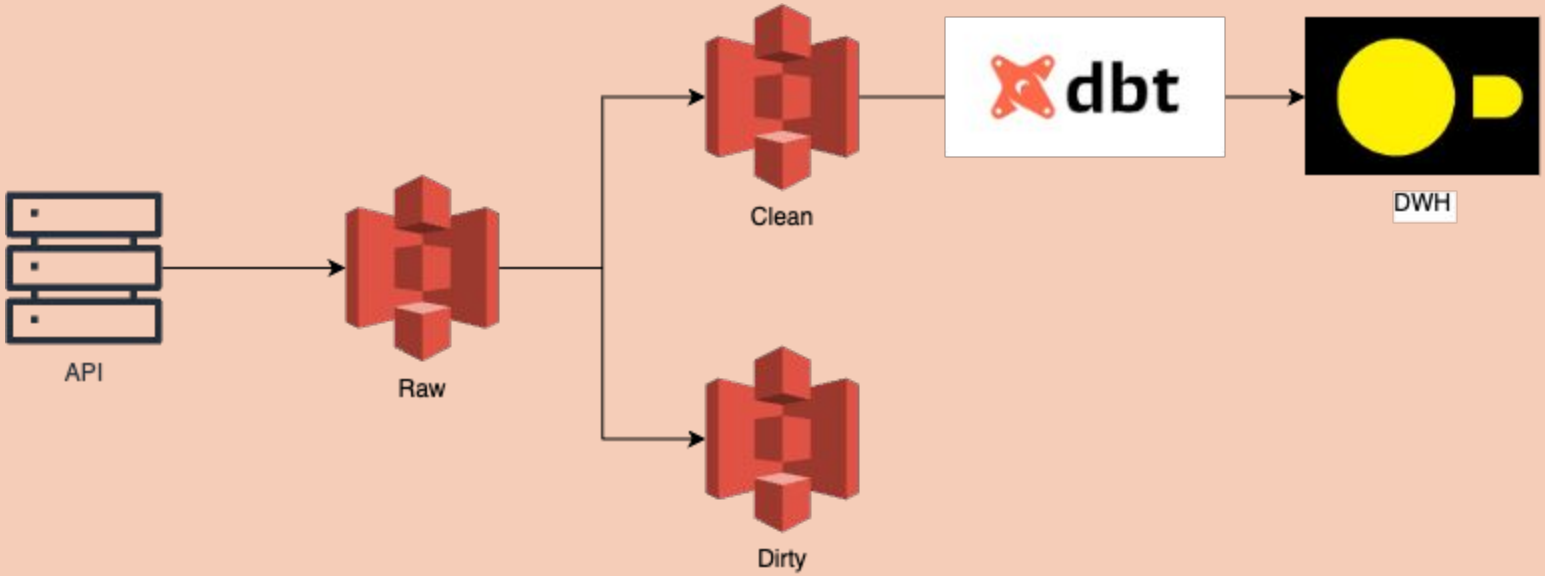
- MLOps, DataOps, data pipeline and ML training tooling see a unification when people realize they are strongly connected

Any MLOps companies you know that deserve a shoutout?

#dataengineering
#machinelearning

Link to the post: https://www.linkedin.com/posts/eczachly_dataengineering-machinelearning-activity-7059281093522272256-xBza

Challenge


API
Raw
Clean
Dirty
dbt
DWH

Shooting myself in the foot

DuckDB

Macbook M1

Apache Airflow

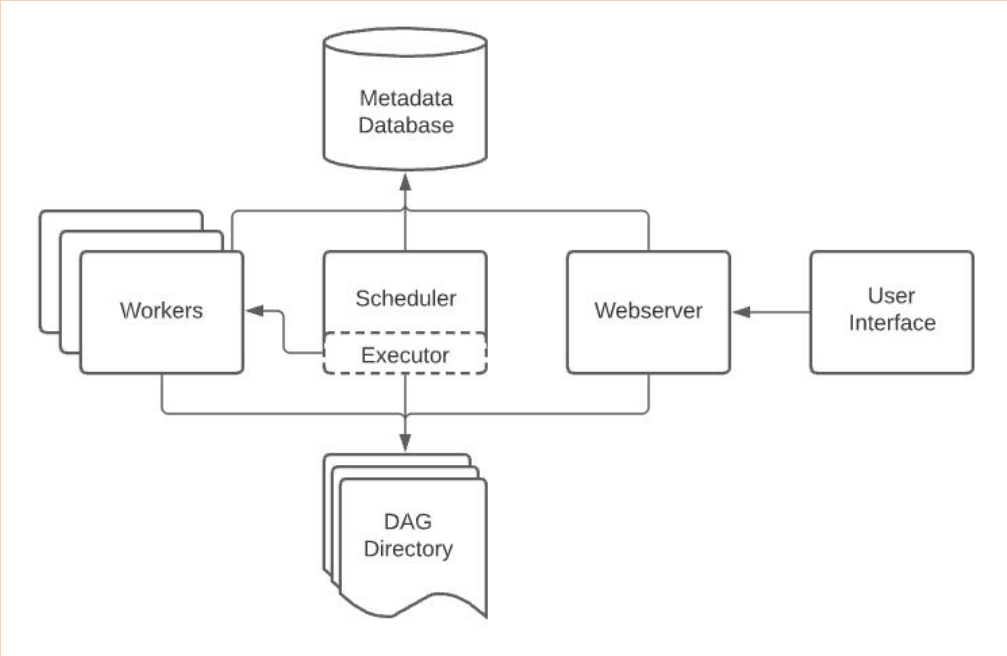
History

- Created at Airbnb in October 2014
- Initial release date to public June 3, 2015
- Apache incubator project in March 2016
- Top-level Apache Software Foundation project in January 2019
- Astronomer is a company that provides SaaS tool of Airflow
- AWS has MWAA, GCP - Cloud Composer, Azure - Data Factory

Adoption

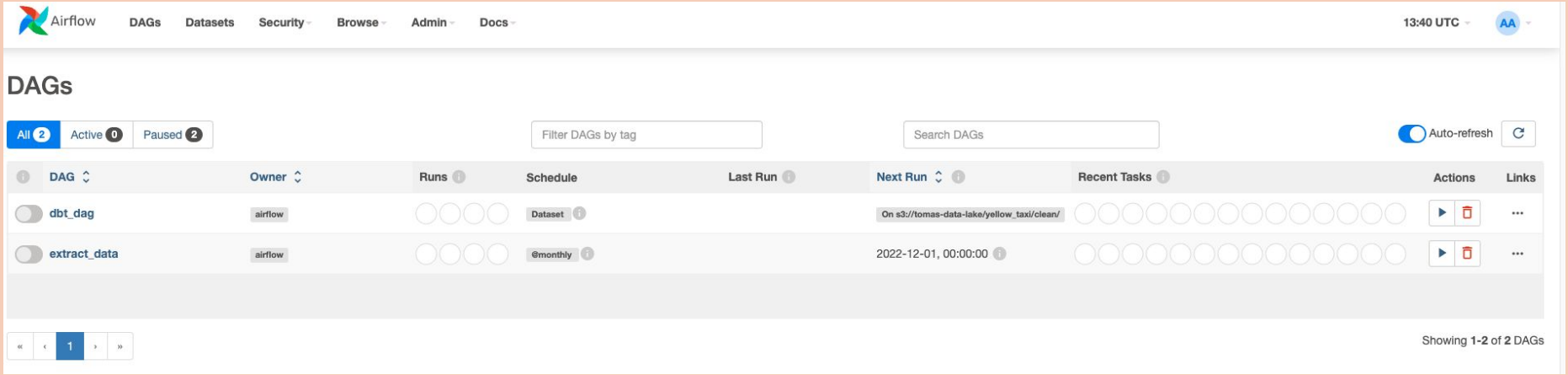
- Apache 2.0 license
- Slack: 31k members
- Github Stars: 30.1k
- Docker pulls: 113M
- Downloads PyPi: 8.4M/month

Architecture



Source: https://airflow.apache.org/docs/apache-airflow/stable/core-concepts/overview.html

User Interface



Airflow

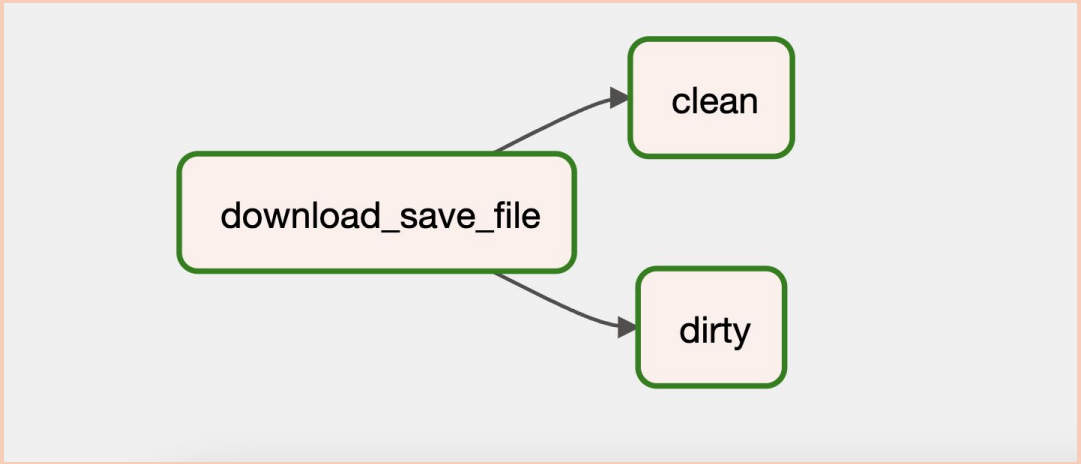
```
#dag
@dag('extract_data',
     default_args=default_args,
     schedule_interval='@monthly',
     start_date=datetime(2022, 11, 30),
     end_date=datetime(2023,1,1), catchup=True)
def extract_data():
```

Airflow

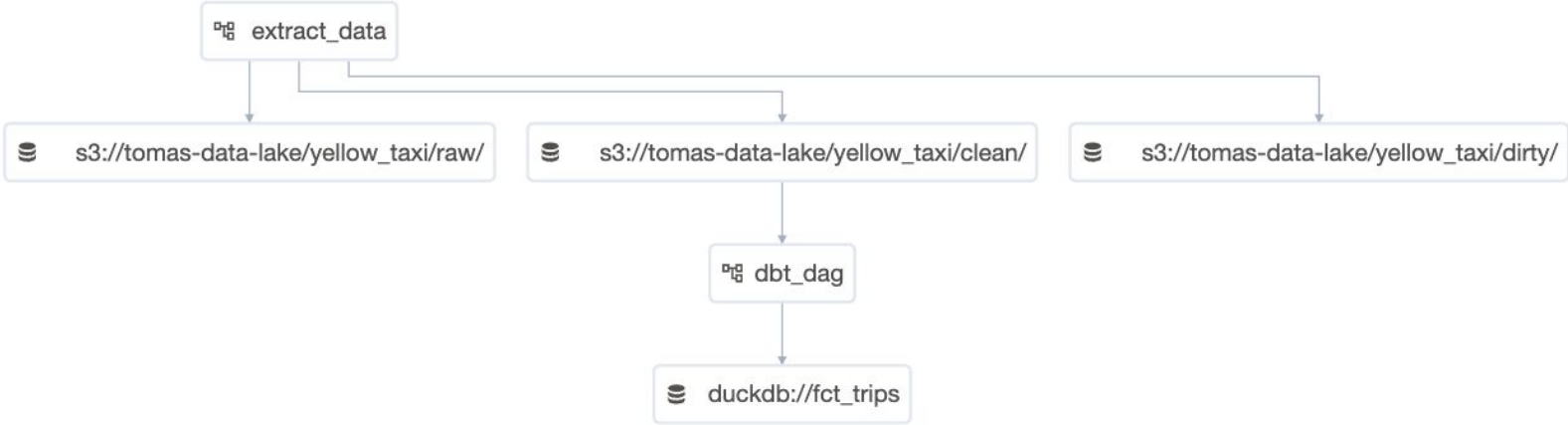
```python
@task(outlets=[download_dir])
def download_save_file(bucket_name, prefix_raw, **kwargs):
    #download file from api
    dt = kwargs['data_interval_start'].format('Y-MM')
    print(dt)
    key_name = f'{prefix_raw}/{dt}/yellow_taxi_{dt}.parquet'
    url = 'https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_tripdata_{dt}.parquet'.format(dt=dt)
    print(url)
    r = requests.get(url)
    print(r)
    data = r.content
    hook = S3Hook('aws_default')
    file_name = f's3://{bucket_name}/{key_name}'

    hook.load_bytes(data, key=key_name, bucket_name=bucket_name)
    return file_name
```

Dag interface



Datasets


extract_data
s3://tomas-data-lake/yellow_taxi/raw/
s3://tomas-data-lake/yellow_taxi/clean/
s3://tomas-data-lake/yellow_taxi/dirty/
dbt_dag
duckdb://fct_trips

Dagster

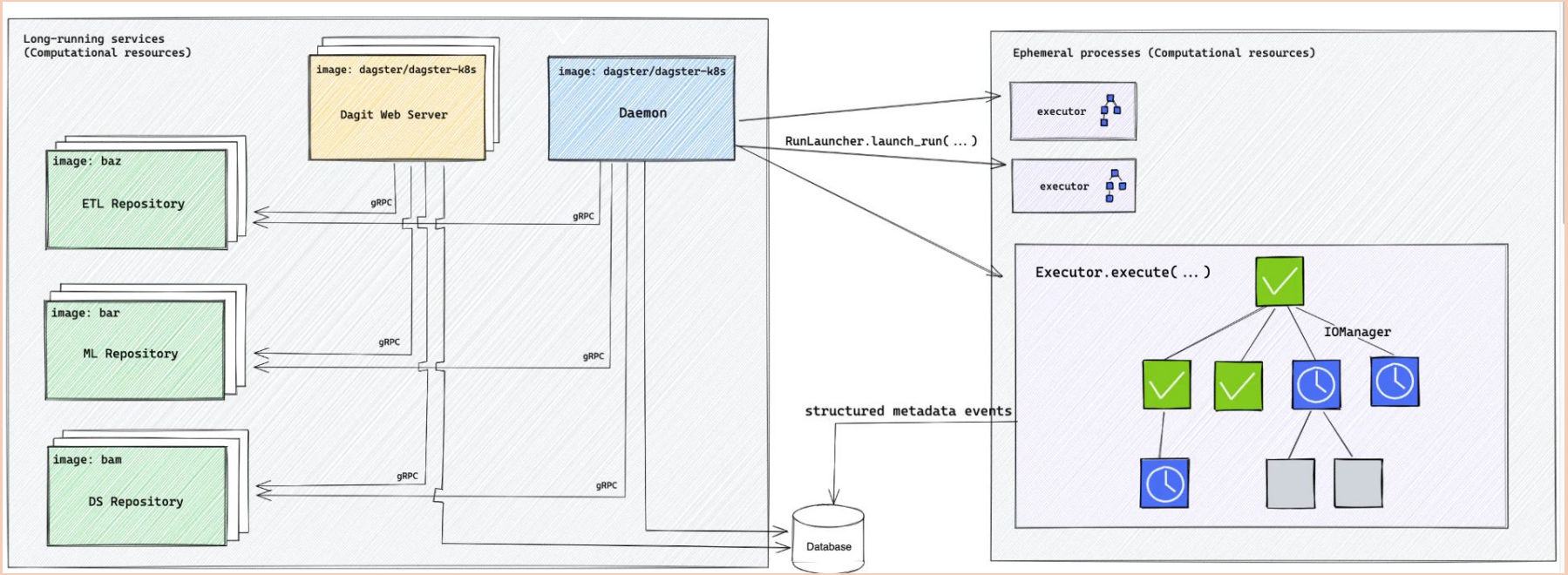
History

- Initial release date to public June 12th, 2018
- v1.0 August 5th, 2022
- Elementl is a company that provides SaaS tool of Dagster

Adoption

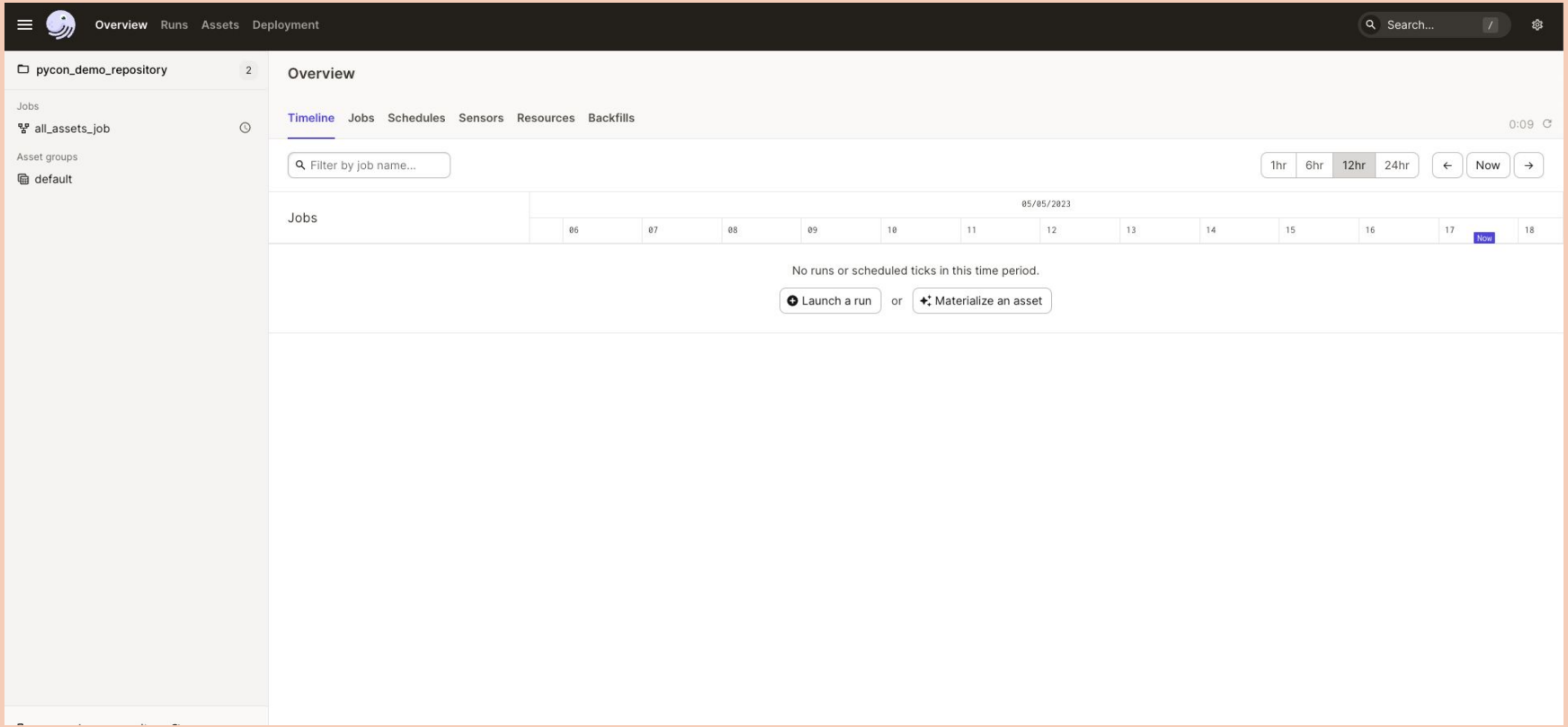
- Apache 2.0 license
- Slack: 6.2k members
- Github Stars: 7.3k
- Docker pulls: ???*
- Downloads PyPi: 526k/month

Architecture



Source: https://docs.dagster.io/deployment/overview

User Interface



Dagster

```
monthly_refresh_schedule = ScheduleDefinition(
    job=define_asset_job(name="all_assets_job"),
    cron_schedule="0 0 5 * *",
)
```

```
defs = Definitions(
    assets=[yellow_tripdata_raw, clean, dirty]+dbt_assets,
    resources={"s3": s3, "dbt": dbt_cli_resource.configured(
        {
            "project_dir":DBT_PROJECT_PATH,
            "profiles_dir": DBT_PROFILES,
        },
    ),},
    schedules=[monthly_refresh_schedule],
)
```

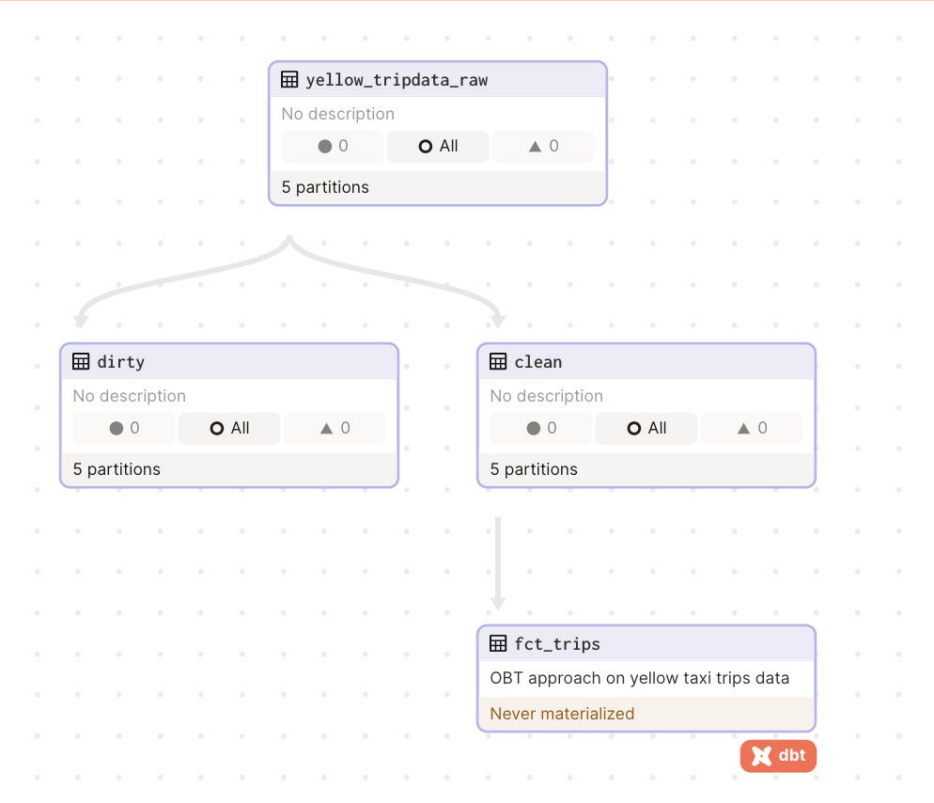
Dagster

```python
partitions_def = MonthlyPartitionsDefinition(start_date="2022-12", fmt="%Y-%m")


@asset(partitions_def=partitions_def, required_resource_keys={"s3"})
def yellow_tripdata_raw(context):
    bucket = "tomas-data-lake"
    prefix_raw = "yellow_taxi/raw"

    partition_date_str = context.asset_partition_key_for_output()
    url = f"https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_tripdata_{partition_date_str}.parquet"
    key_name = (
        f"{prefix_raw}/{partition_date_str}/yellow_taxi_{partition_date_str}.parquet"
    )
    r = requests.get(url)
    data = r.content
    s3_client = context.resources.s3
    s3_client.put_object(Bucket=bucket, Key=key_name, Body=data)
```

Dagster


yellow_tripdata_raw
No description
0
All
0
5 partitions
dirty
No description
0
All
0
5 partitions
clean
No description
0
All
0
5 partitions
fct_trips
OBT approach on yellow taxi trips data
Never materialized
dbt

Mage

History

- Company established December 1st 2020
- First release June 9th 2022

Adoption

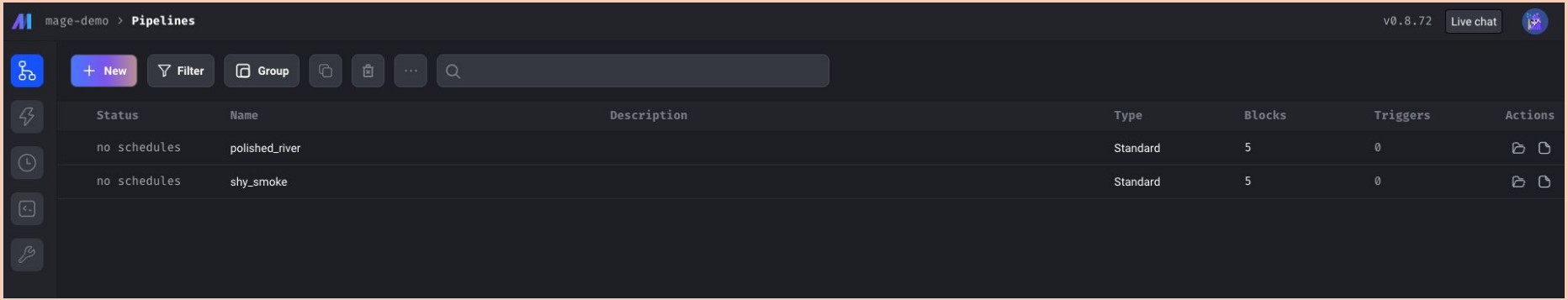
- ??? license
- Slack: 1.5k members
- Github Stars: 4.1k
- Docker pulls: 132k
- Downloads PyPi: 9k/month

Architecture

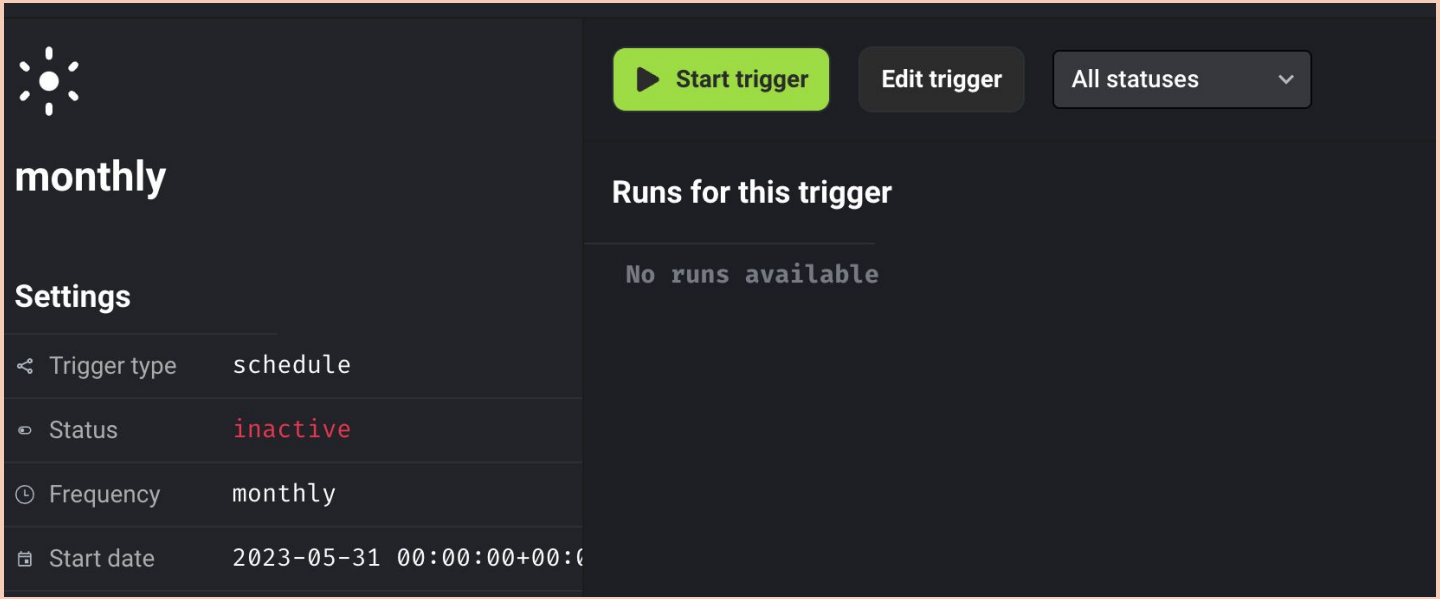
- No publicly available information. My assumptions only!

- Based on docker-compose
  - Metadata DB
  - Server (Web and Backend together)
  - Executors are created similarly to Airflow and Dagster

User Interface



Mage

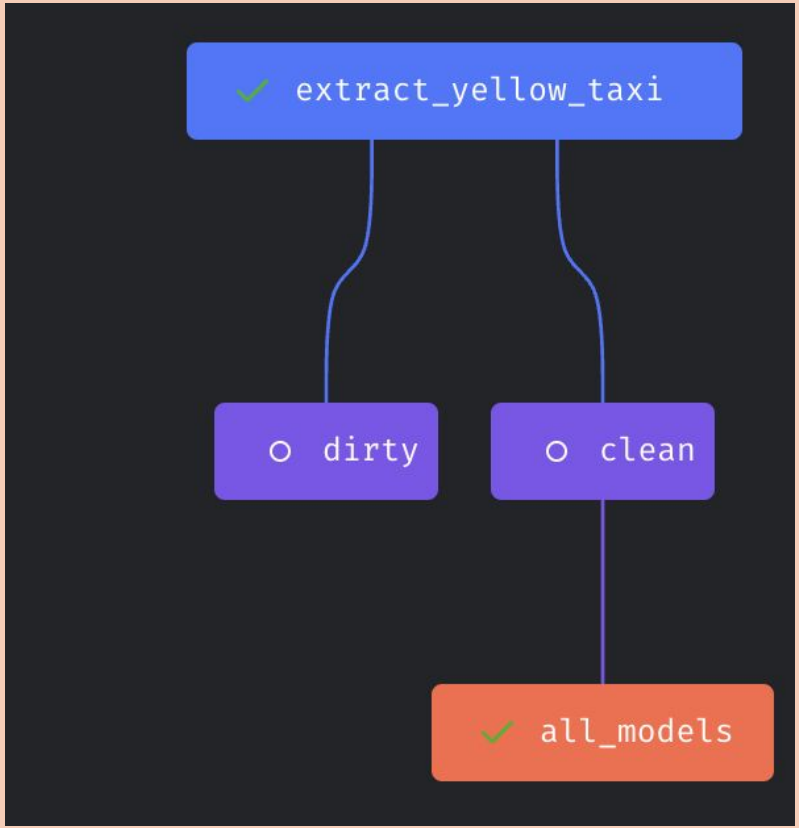

Start trigger
Edit trigger
All statuses
monthly
Runs for this trigger
No runs available
Settings
Trigger type
schedule
Status
inactive
Frequency
monthly
Start date
2023-05-31 00:00:00+00:0

Mage

```python
@data_loader
def load_data_from_api(*args, **kwargs):
    bucket = "tomas-data-lake"
    prefix_raw = "yellow_taxi/raw"
    dt = kwargs['execution_date'].strftime('%Y-%m')
    print(dt)
    url = f"https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_tripdata_{dt}.parquet"
    print(url)
    key_name = (
        f"{prefix_raw}/{dt}/yellow_taxi_{dt}.parquet"
    )
    print(key_name)
    r = requests.get(url)
    data = r.content

    boto3.client('s3',
        aws_access_key_id=os.environ['AWS_ACCESS_KEY_ID'],
        aws_secret_access_key=os.environ['AWS_SECRET_ACCESS_KEY']
        ).put_object(Bucket=bucket, Key=key_name, Body=data)
```

Pipeline



Comparisons

Results

Task	Apache Airflow (Total time: 1min 3s)	Dagster (Total time: 1min 18s)	Mage (Total time: 46s)
Download and store to S3	10s	9s	12s
Clean dataset	23s	22s	15s
Dirty dataset	14s	11s	7s
fct_trips	15s	16s	12s

Same python code was executed in similar times. Bottleneck between different times. Mostly visible in dagster

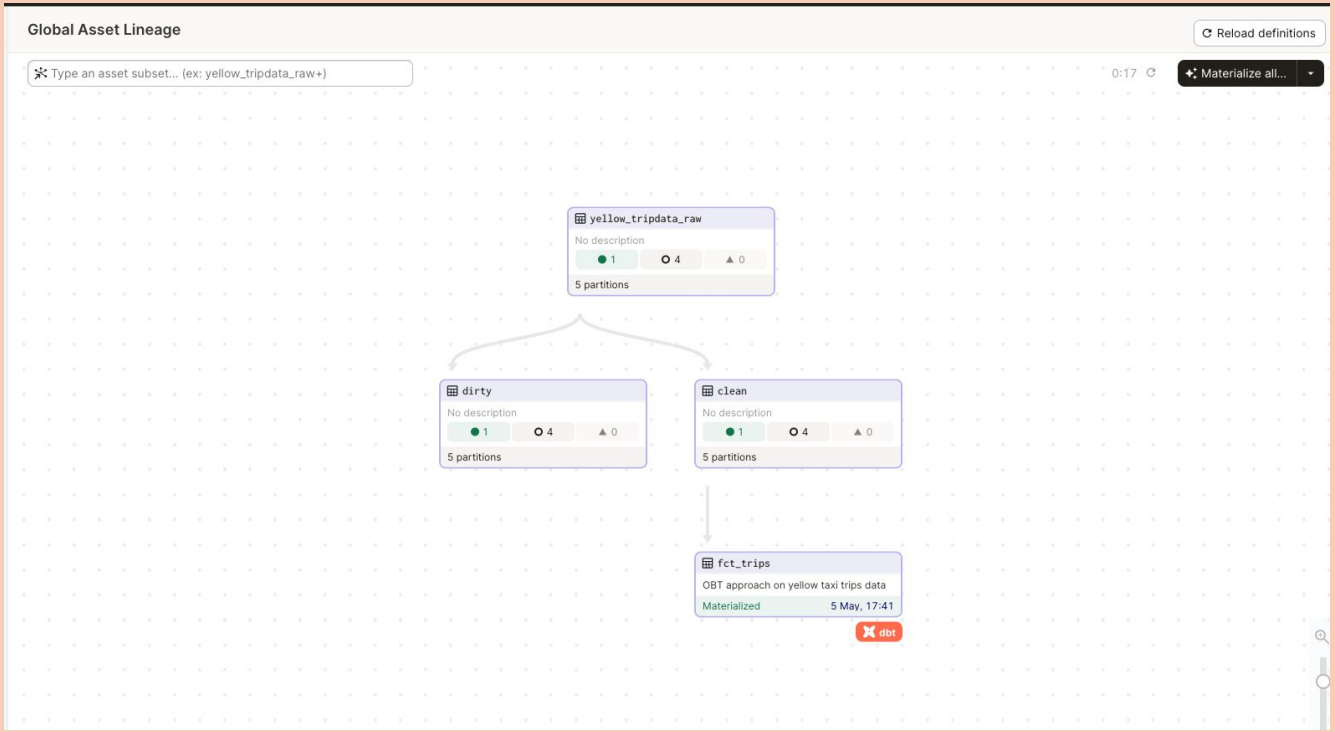
Thoughts and Summary

Dagster

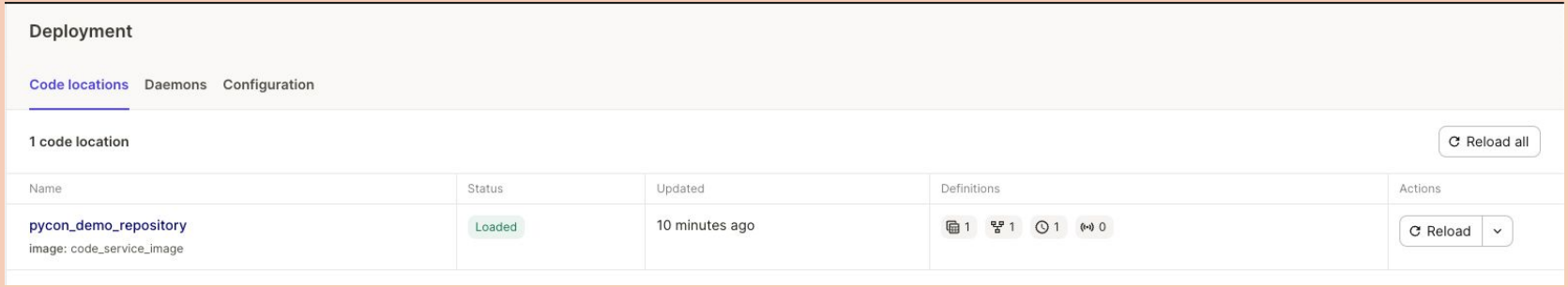
Quick thoughts

- Definitions Similar to Airflow (Ops = Task)
- Feels more Pythonic
- Pre-made migrationary scripts/steps from Airflow
- Feels more clean and lightweight
- Copilot/ChatGPT can’t help much
- Harder to pick up deployment and folder structure (Maybe it’s me?)

Data Lineage/Asset approach



Multi-repository deployments



Mage

Quick thoughts

- Notebooks style
- Hard to pick-up things if you’re an IDE person
- Dependencies drag and drop!
- Copilot/ChatGPT can’t help much
- A bit buggy, but quick support from creators
- dbt things has to be specifically placed inside (little customization)
- Has Streaming pipelines (connects to Kafka!) out of the box (not tested)
- Works and then suddenly doesn’t. Not mature enough yet

Is it actually the end for Apache Airflow?

Not Yet

- Big community
- A lot of prebuilt Operators/Sensors/Hooks
- Battle tested
- Adopts to the changing needs

Questions?