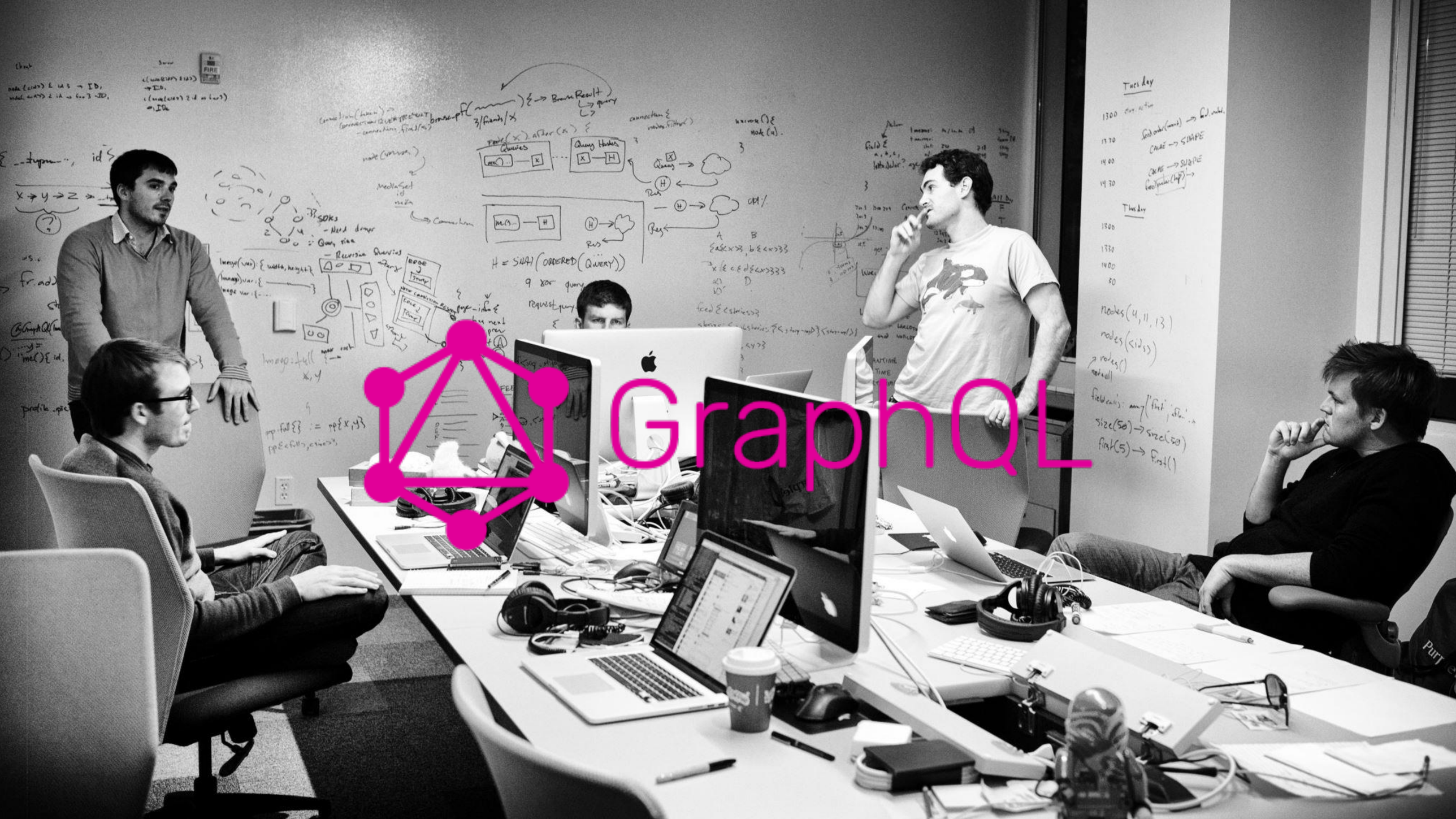
DAGSTER

Nick Schrock
Founder, Elementl
@schrockn


GraphQL

“Our data is totally broken”

“Our data is totally broken”

- We don’t know where our data comes from
- We don’t know what it means
- We cannot reliably process and test it
- Our engineers don’t want to deal with it
- It isn’t “fun.” It isn’t “sexy.”

DATA

Data Scientist: The Sexiest Job of the 21st Century

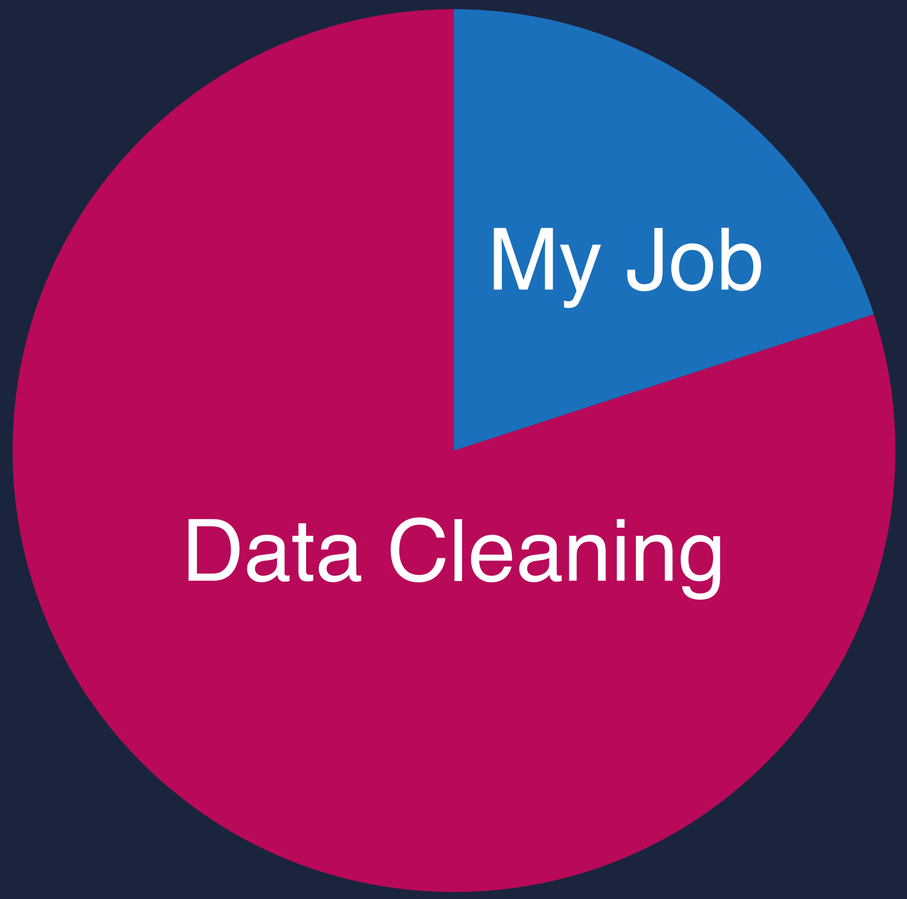
by Thomas H. Davenport and D.J. Patil

What do data scientists actually do on their day-to-day? Career (self.datascience)

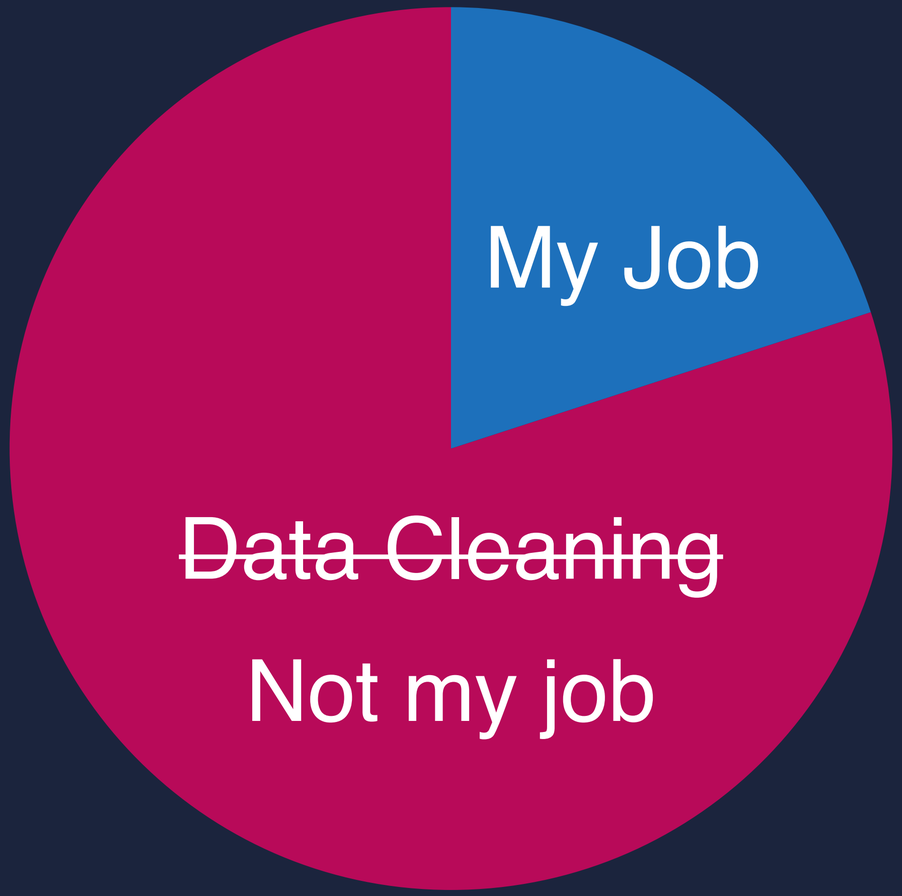
submitted 7 hours ago by giokrist to r/datascience

40 comments share save hide give award report crosspost

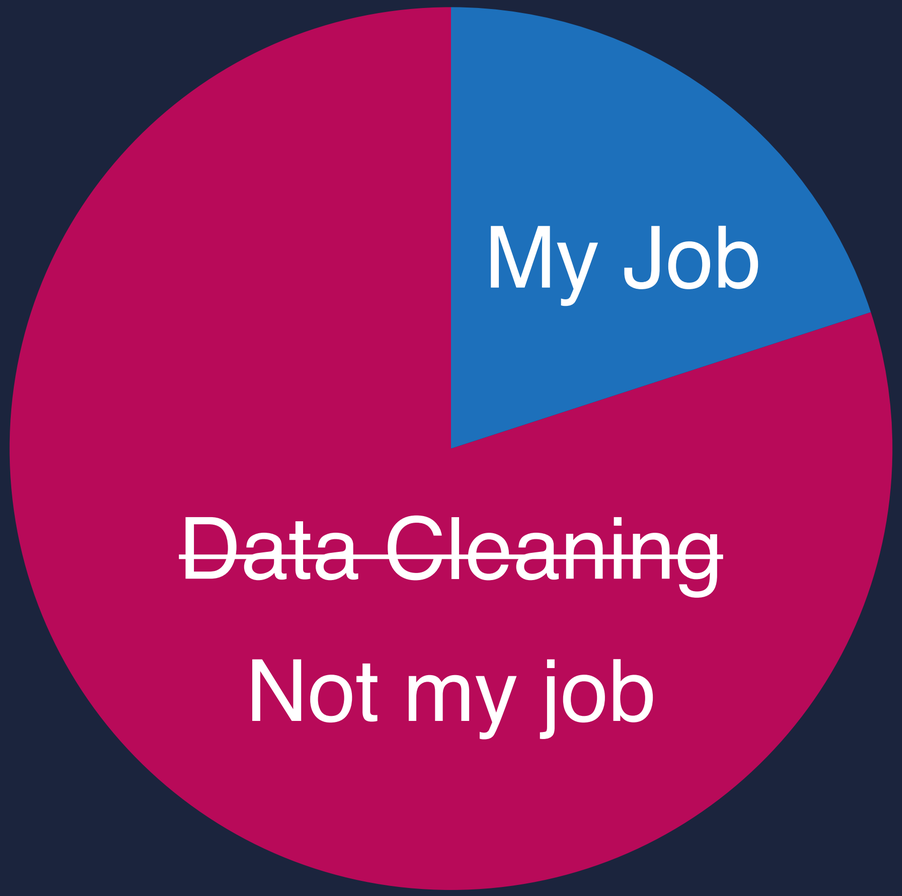
WHAT THEY SAY



WHAT THEY MEAN



WHAT THEY MEAN



WHAT THEY MEAN

- Rolling their own infrastructure
- Repeated work
- Maintaining unreliable processe

FAILURE IS THE NORM

Big data strategies disappoint with 85 percent failure rate

BY JAMES WALKER NOV 23, 2017 IN TECHNOLOGY LISTEN | PRINT

Big data projects aren't delivering the transformations companies had anticipated. 85% of all big data strategies are failing due to a combination of challenges, including the problems presented by legacy technologies and pre-existing corporate biases.

Engineers: I don’t want to touch it.

Data scientist: I waste most of my time.

Business Leader: Failure is the norm.

2009: UI development is awful

- I spend 80% of my time fighting the browser

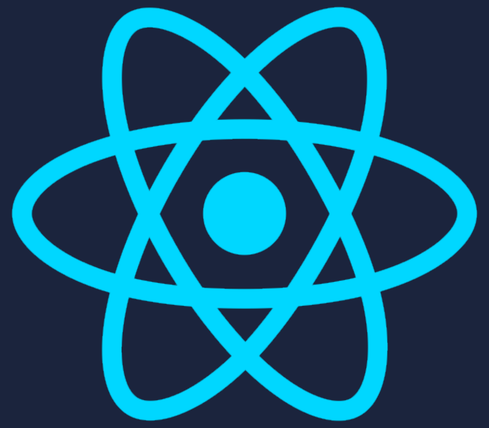
2009: UI development is awful

- I spend 80% of my time fighting the browser
- We can’t change our UI–there’s no testing
- It breaks all the time.
- Our engineers don’t want to touch it

2019: A (UI) world transformed

Browsers did get better.

But it was the software abstractions that proved decisive.



Scripts → Full applications

React acknowledged complexity

It respected the discipline

React → Frontend Applications

Dagster → Data Applications

PRINCIPLEs

- Solves a real problem
- Incremental adoption path
- Preserve tools that work
- Immediate value and productivity gains


WHAT WOULD YOU SAY
YOU DO HERE?

Data [illegible]tions

Graphs of functional computations
that produce and consume data assets

```
> pip install dagster
```

pagerank.py

pagerank.py

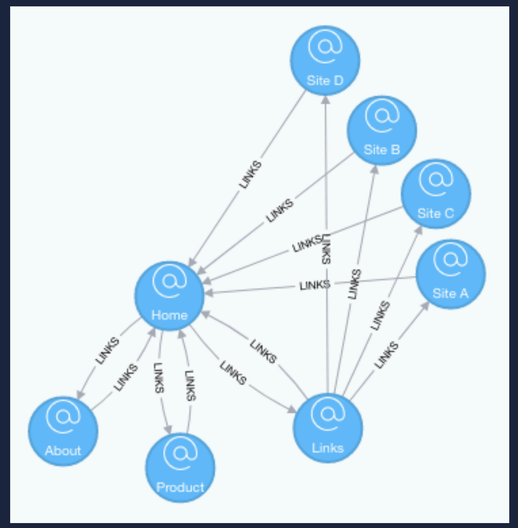
1

schrockn/screencast-notebooks* Ln 1, Col 1 Spaces: 4 UTF-8 LF Python

```
(dagster-3.7.1) [schrockn@mbp ~/code/dagster/examples/pyspark_pagerank (schrockn/screencast-notebooks)]$ dagit -f pagerank.py -n define_pipeline
```

DAGSTER CONCEPTS

- Solid: A unit of functional computation
- Pipeline: A DAG of solids



Page Rank on Spark

original.py × pagerank.py

original.py ▸ ...

```python


if __name__ == "__main__":
    if len(sys.argv) != 3:
        print("Usage: pagerank <file> <iterations>", file=sys.stderr)
        sys.exit(-1)

    print(
        "WARN: This is a naive implementation of PageRank and is given as "
        + "an example!\nPlease refer to PageRank implementation provided "
        + "by graphx",
        file=sys.stderr,
    )

    # Initialize the spark context.
    execute_pagerank(sys.argv[1], int(sys.argv[2]))

```

schrockn/screencast-notebooks 3↓ 7↑ Ln 86, Col 10 Spaces: 4 UTF-8 LF Python

DAGSTER

Select a Pipeline...

0.3.5

No pipeline selected

Select a pipeline in the navbar

DAGSTER CONCEPTS

- Solid
  - Inputs: Inputs are the data
  - Config: Config modifies how data is computed
- Pipeline

```python
@solid(
    inputs=[InputDefinition('path', Path)],
    config_field=Field(Dict({'iterations': Field(Int)})),
)
def execute_pagerank(context, path):
    spark = SparkSession.builder.appName("PythonPageRank").getOrCreate()
    lines = spark.read.text(path).rdd.map(lambda r: r[0])
    links = lines.map(parseNeighbors).distinct().groupByKey().cache()
    ranks = links.map(lambda url_neighbors: (url_neighbors[0], 1.0))
    iterations = context.solid_config['iterations']
```

```python
@solid(inputs=[InputDefinition('path', Path)])
def load_pagerank_data(_context, path):
    spark = SparkSession.builder.appName("PythonPageRank").getOrCreate()
    lines = spark.read.text(path).rdd.map(lambda r: r[0])
    return lines.map(parseNeighbors)


@solid(
    inputs=[InputDefinition('links')],
    config_field=Field(Dict({'iterations': Field(Int)})),
)
def execute_pagerank(context, links):
    spark = SparkSession.builder.appName("PythonPageRank").getOrCreate()
    cached_links = links.distinct().groupByKey().cache()
    ranks = cached_links.map(lambda url_neighbors: (url_neighbors[0], 1.0))
    iterations = context.solid_config['iterations']
```

```python
@solid(inputs=[InputDefinition('path', Path)])
def load_pagerank_data(context, path):
    # ...

@solid(
    inputs=[InputDefinition('links')],
    config_field=Field(Dict({'iterations': Field(Int)})),
)
def execute_pagerank(context, links):
    # ...
```

```python
@solid(inputs=[InputDefinition('path', Path)])
def load_pagerank_data(context, path):
    # ...

@solid(
    inputs=[InputDefinition('links')],
    config_field=Field(Dict({'iterations': Field(Int)})),
)
def execute_pagerank(context, links):
    # ...
```

```python
def define_pipeline():
    return PipelineDefinition(
        name='pagerank',
        solids=[load_pagerank_data, execute_pagerank],
        dependencies={
            'execute_pagerank': {
                'links': DependencyDefinition('load_pagerank_data')
            }
        },
    )
```

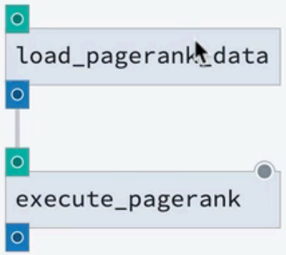
DAGSTER

pagerank

Explore Execute Runs

0.3.5

Select a Solid... Filter...



Info Types

Pipeline

pagerank

DESCRIPTION

CONTEXTS

default

```
/* A configuration dictionary with typed
   fields */
{
  log_level?: log_level
}
```

DAGSTER CONCEPTS

- Solid
  - Inputs & Config
- Pipeline
- Dependencies

Before:

```
@solid(inputs=[InputDefinition('path', Path)])
def load_pagerank_data(_context, path):
    # Initialize the spark context.
    spark = SparkSession.builder.appName("PythonPageRank").getOrCreate()

    # two urls per line with space in between)
    lines = spark.read.text(path).rdd.map(lambda r: r[0])
```

After:

```
@solid(inputs=[InputDefinition('path', Path)])
def load_pagerank_data(context, path):
    # two urls per line with space in between)
    lines = context.resources.spark.read.text(path).rdd.map(lambda r: r[0])
```

```python
from dagster_pyspark import spark_session_resource
```

```python
def define_pyspark_pagerank():
    return PipelineDefinition(
        name='pyspark_pagerank',
        context_definitions={
            'local': PipelineContextDefinition(
                resources={'spark': spark_session_resource}
            )
        },
```

Chrome File Edit View History Bookmarks People Window Help 100% Sun 3:23 PM

Dagit | Dagit | Dagit | Dagit | Dagit

127.0.0.1:3000/pagerank/explore

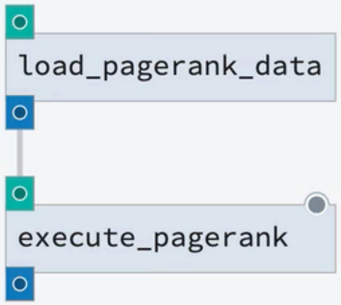
DAGSTER

pagerank

Explore Execute Runs

0.3.5

Select a Solid... Filter...



Info Types

Pipeline

pagerank

DESCRIPTION

CONTEXTS

local

```
/* A configuration dictionary with
   typed fields */
{
  log_level?: log_level
}
```

spark

```
/* A configuration dictionary with
{
  spark_conf?: {
    spark?: {
      app?: {
        /* Application Properties:
```

DAGSTER CONCEPTS

- Solid
  - Inputs & Config
- Pipeline
- Dependencies
- Context
  - Logging: Structured Logging
  - Resources: Connections, Services, Etc

Beautiful, High-Quality Tools

Dagit

Editor | DAG View | Console

API

DAGSTER

Python library

Dagster Libraries and Integrations

PySpark

Graph of Functional Computation

API

- Queryable and Introspectable
- Operable
  - Executable and Configurable
- Monitorable
  - Logging and Live Subscriptions

Dagster: a platform for building tools

Beautiful, High-Quality Tools

Dagit

Editor | DAG View | Console

API

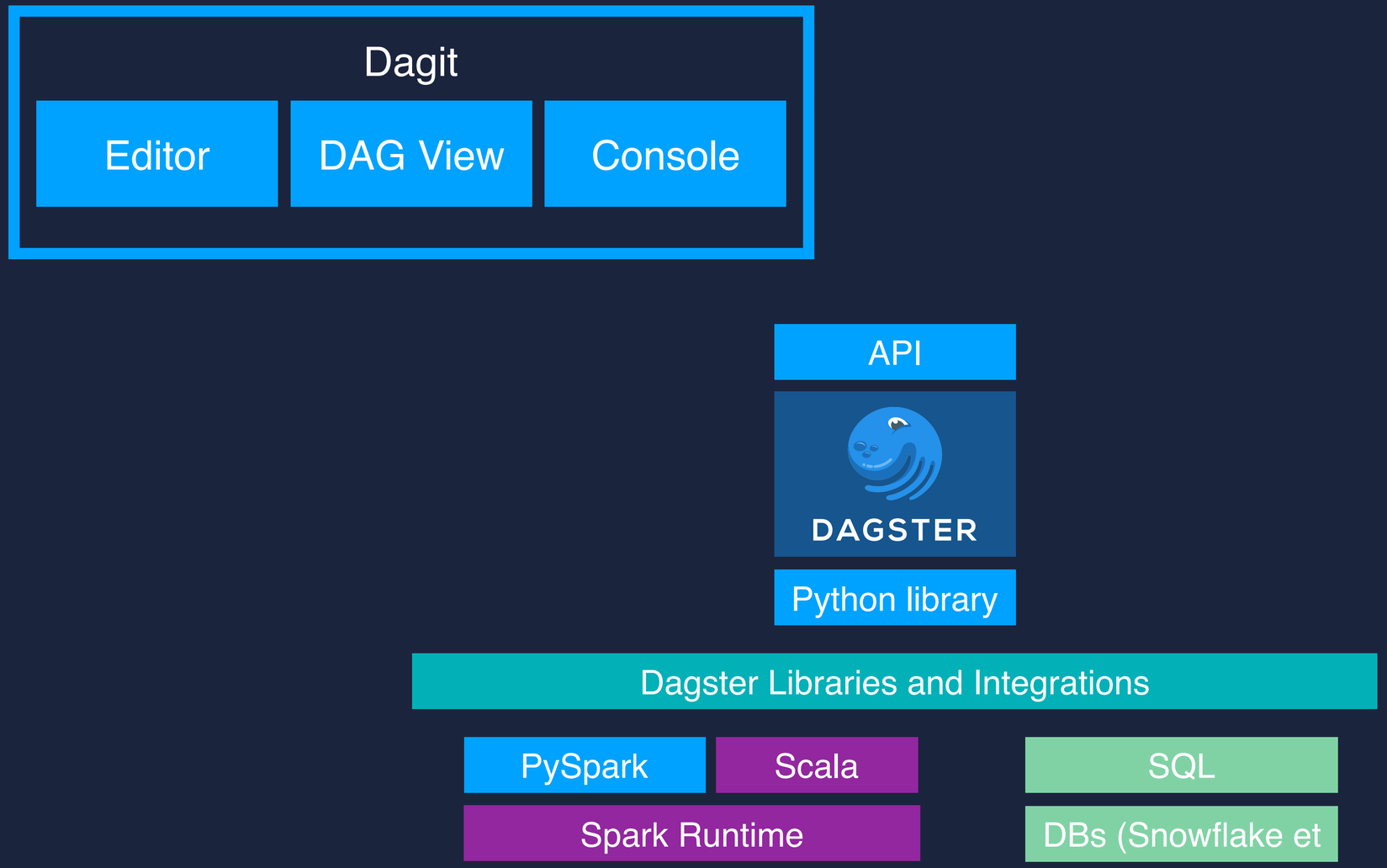
DAGSTER

Python library

Dagster Libraries and Integrations

PySpark

Beautiful, High-Quality Tools



```
event_ingest = SparkSolidDefinition(
    name='event_ingest',
    main_class='io.dagster.events.EventPipeline',
    description='Ingest events from JSON to Parquet',
)

snowflake_load = SnowflakeLoadSolidDefinition(
    'snowflake_load', table='events'
)

return PipelineDefinition(
    name='event_ingest_pipeline',
    solids=[download_from_s3, gunzipper, event_ingest, snowflake_load],
    dependencies={
```

DAGSTER

Select a Pipeline...

0.3.5

No pipeline selected

Select a pipeline in the navbar

- Open Source, Python Library
- Multi-lingual integration
- Beautiful Tooling

jupyter

O'REILLY

jupytercon

Brought to you by NumFOCUS Foundation and O'Reilly Media Inc.

I don't like notebooks.

Joel Grus

Allen Institute for Artificial Intelligence

jupytercon.com

#JupyterCon

Beyond Interactive: Notebook Innovation at Netflix

Netflix Technology Blog Follow

Aug 16, 2018 · 13 min read

By Michelle Ufford, M Pacer, Matthew Seal, and Kyle Kelley

Notebooks have rapidly grown in popularity among data scientists to become

---

README.md

papermill

build passing | codecov 90% | docs passing | highlight binder | cli binder | code style black

papermill is a tool for parameterizing, executing, and analyzing Jupyter Notebooks.

Papermill lets you:

- **parameterize** notebooks
- **execute** notebooks

DAGSTER

Select a Pipeline...

0.3.5

No pipeline selected

Select a pipeline in the navbar

jupyter Geographic_Distribution_of_Recent_Events Last Checkpoint: a few seconds ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted dagster

Run Code

In [18]:

```
import dagstermill as dm
from event_pipeline_demo.repository import define_repo
dm.register_repository(define_repo())
import pandas as pd
```

In [19]: parameters

```
df = pd.read_csv('table_to_df.csv')
```

In [20]:

```
df.head()
```

Out[20]:

	latitude	longitude	timestamp
0	46.18396	6.10237	2019-01-02 15:59:59
1	37.74615	-25.66689	2019-01-02 15:59:57
2	-30.60106	-71.19901	2019-01-02 15:59:54
3	35.85000	117.70000	2019-01-02 15:59:52
4	6.03333	37.55000	2019-01-02 15:59:47

In [21]:

```
import os
import geopandas as geo
import matplotlib.pyplot as plt
import pandas as pd
```

DAGSTER

event_ingest_pipeline

Explore Execute Runs

0.3.5

Select a Solid... Filter...

download_from_s3

gunzipper

event_ingest spark

snowflake_load sql

table_to_df

geo_events ipynb

Info Types

Pipeline

event_ingest_pipeline

DESCRIPTION

CONTEXTS

default

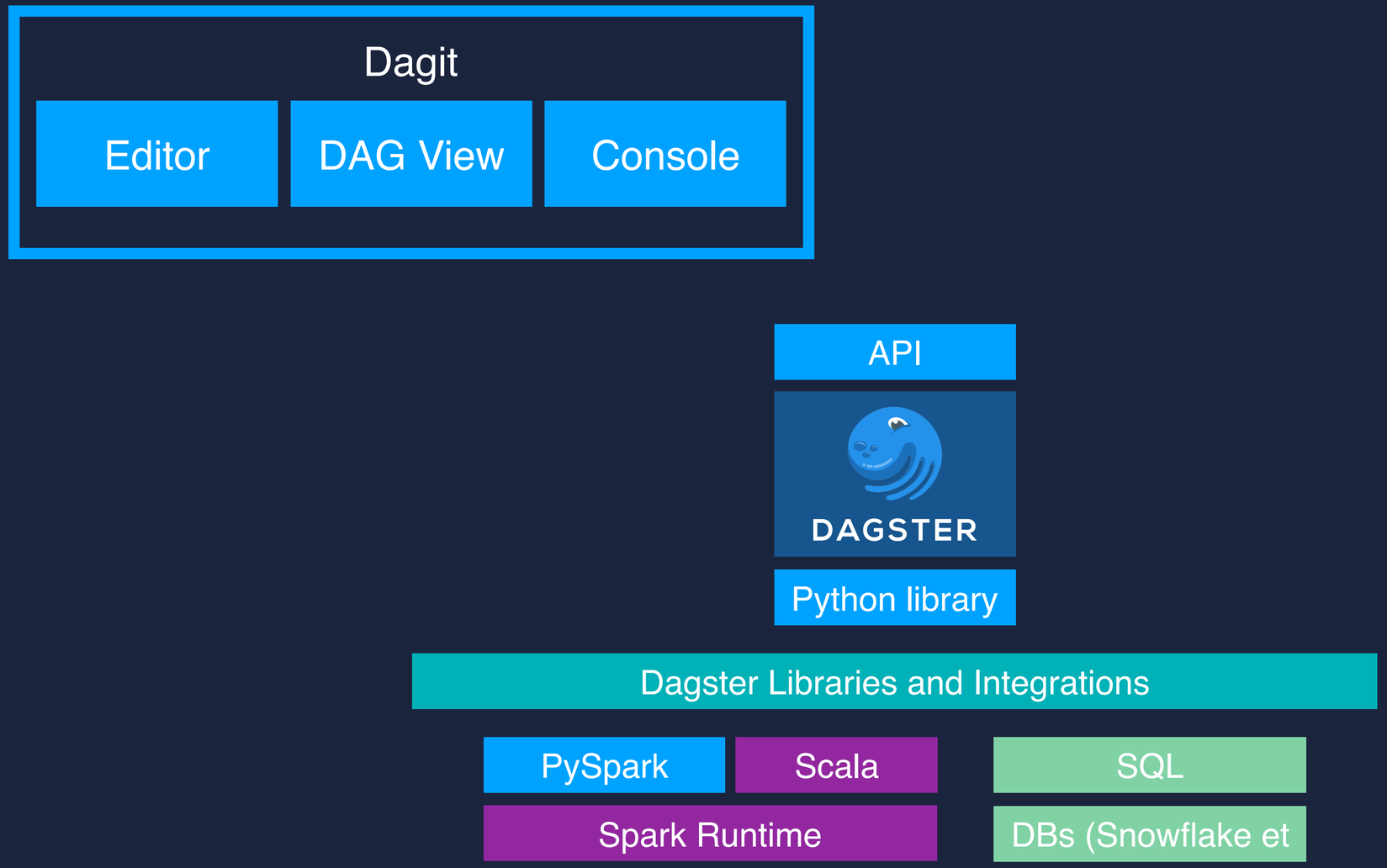
```
/* A configuration dictionary with typed
   fields */
{
  log_level?: log_level
}
```

iTerm2 Shell Edit View Session Scripts Profiles Toolbelt Window Help 100% Tue 7:13 PM

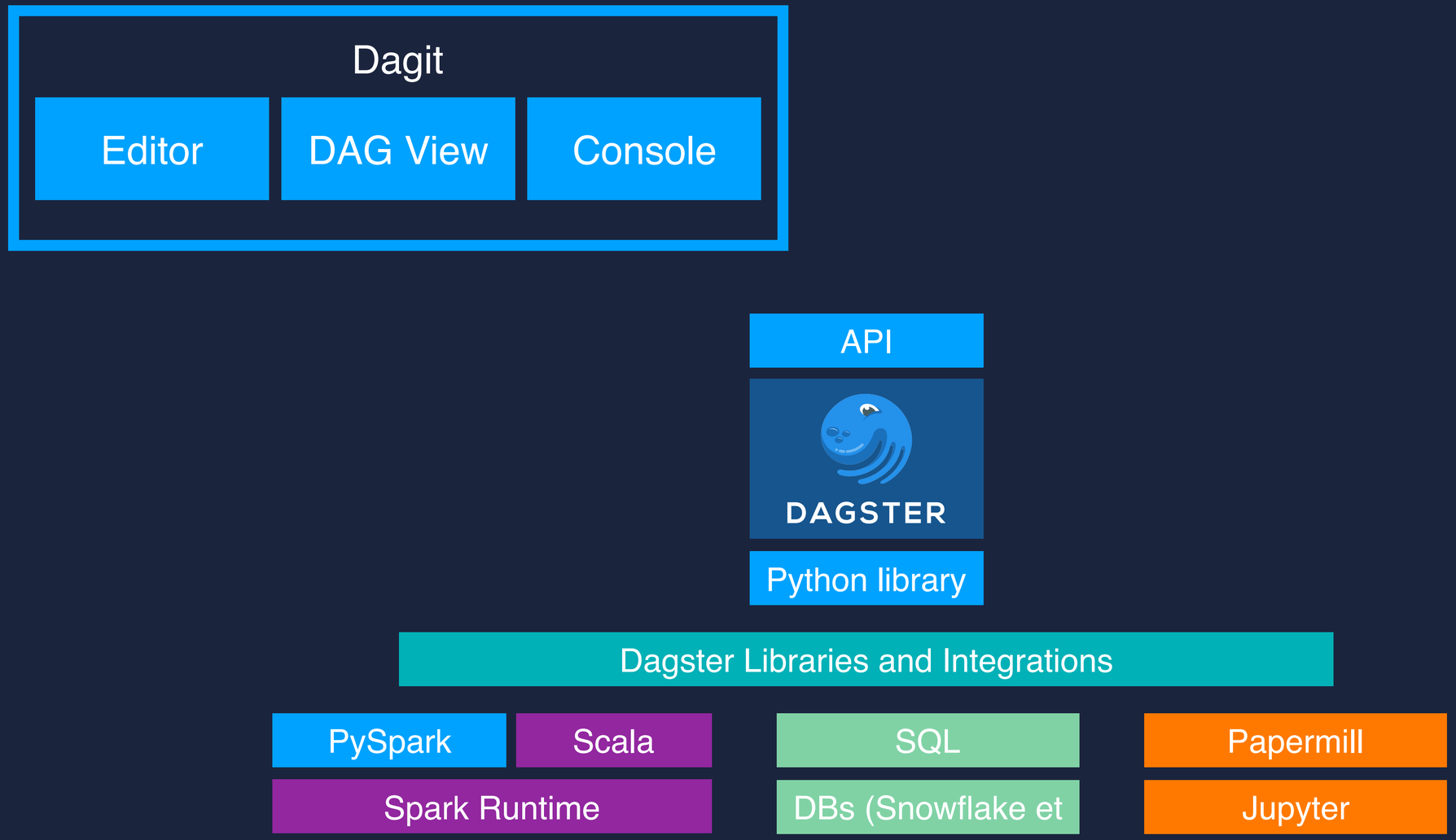
2. bash

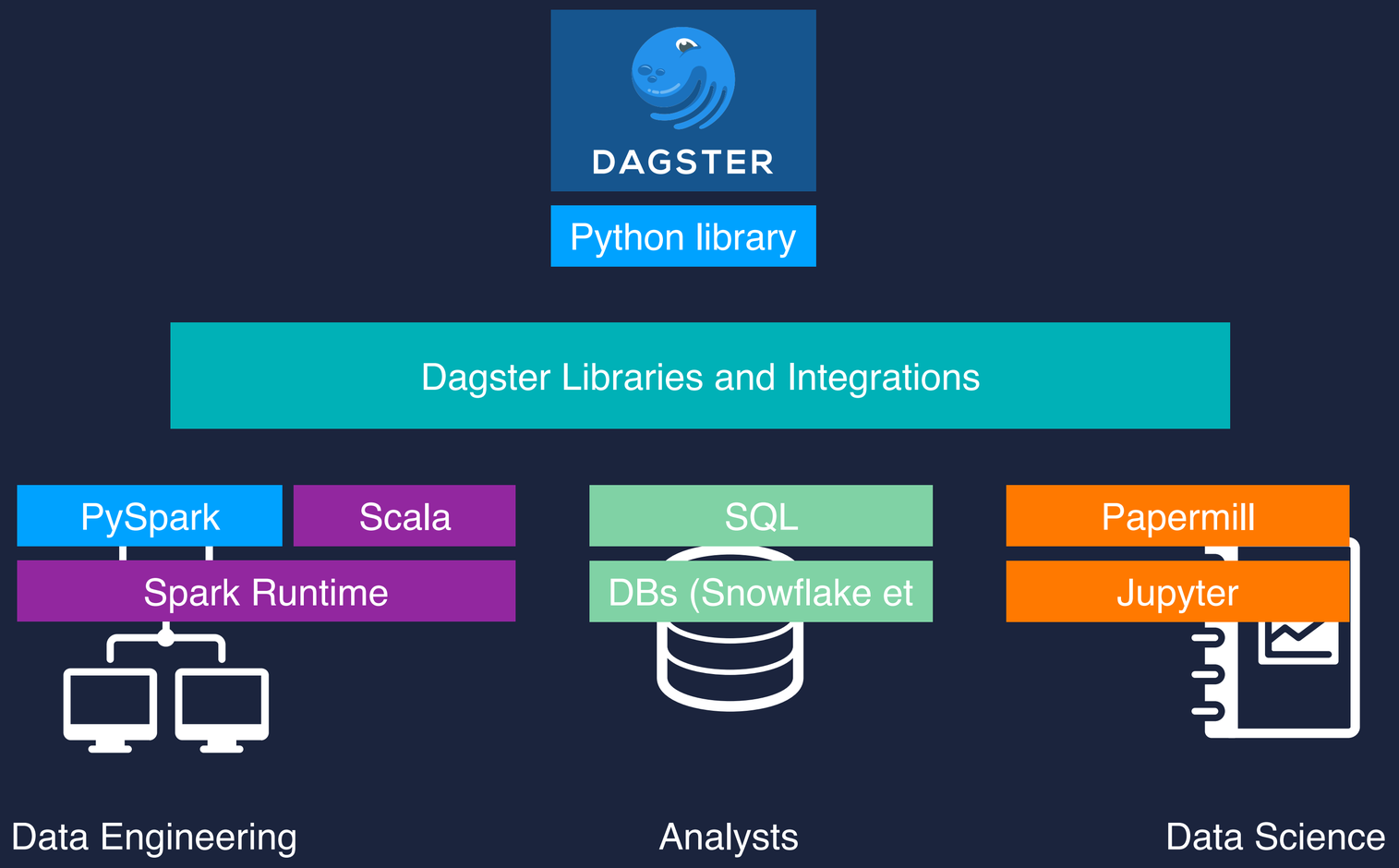
```
(dagster-3.7.1) [schrockn@mbp ~/code/dagster (schrockn/airflowize *)]$
```

Beautiful, High-Quality Tools

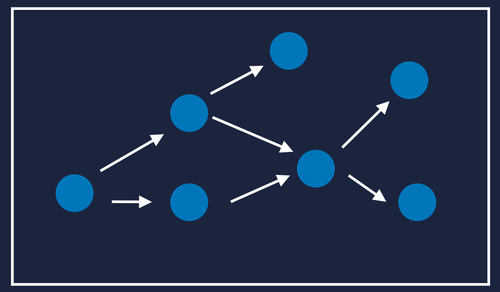


Beautiful, High-Quality Tools




DAGSTER
Python library
Dagster Libraries and Integrations
PySpark
Scala
Spark Runtime
SQL
DBs (Snowflake et
Papermill
Jupyter
Data Engineering
Analysts
Data Science

Data Application: A Graph of Computations



DAGSTER

Dagster Libraries and Integrations

Data Engineering

Analysts

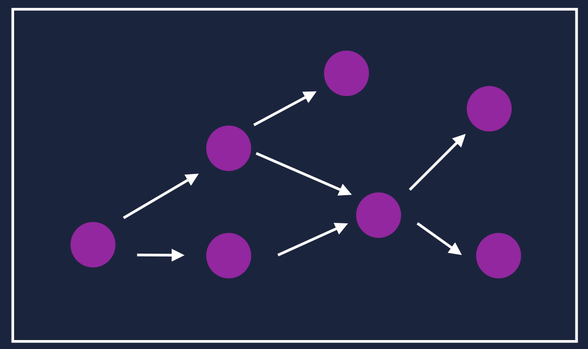
Data Science

API

dagit

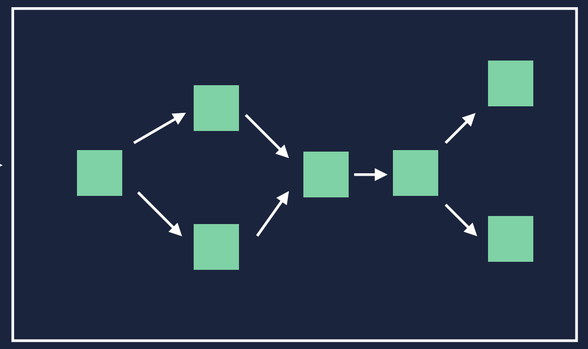
dagster cli

Local Executor



dagster-

Airflow



```python
from dagster.utils import load_yaml_from_globs
from dagster_airflow.factory import make_airflow_dag
from event_pipeline_demo.pipelines import define_event_ingest_pipeline

PATH_TO_CONFIG_YML = '/Users/schrockn/event-pipeline-demo/environments/*.yml'

dag, steps = make_airflow_dag(
    define_event_ingest_pipeline(),
    env_config=load_yaml_from_globs(PATH_TO_CONFIG_YML),
    dag_id='event_ingest_pipeline',
    dag_description='A demo Airflow DAG corresponding to the event ingest pipeline.',
    dag_kwargs=None,
    op_kwargs=None,
)
```

iTerm2 Shell Edit View Session Scripts Profiles Toolbelt Window Help 100% Tue 7:24 PM

2. bash

```
(dagster-3.6.7) [schrockn@mbp ~/code/dagster (schrockn/airflowize *)]$ airflow webserver
```

DATA ENGINEERING

- An emerging discipline
- At an inflection point

Scripts → Data Applications

ELEGANT PROGRAMMING MODEL

NEW, BEAUTIFUL TOOLING

FLEXIBLE AND INCREmENTAL

FLEXIBLE AND INCREmENTAL

- Use your tools
- Preserve your code
- Deploy to your infrastructure
- Adopt incrementally

And there is a ton of work to do

TEAM

Max Gasner

Nate Kupp

Alex Langenfeld

THANK YOU

Ben Gotow

Mikhail Novikov

Uma Roy

THANK YOU

Abe Gong

Superconductive Health

https://github.com/dagster-io/dagster

Join the team. Partner with us.

https://elementl.com

schrockn@elementl.com

@schrockn

DAGSTER