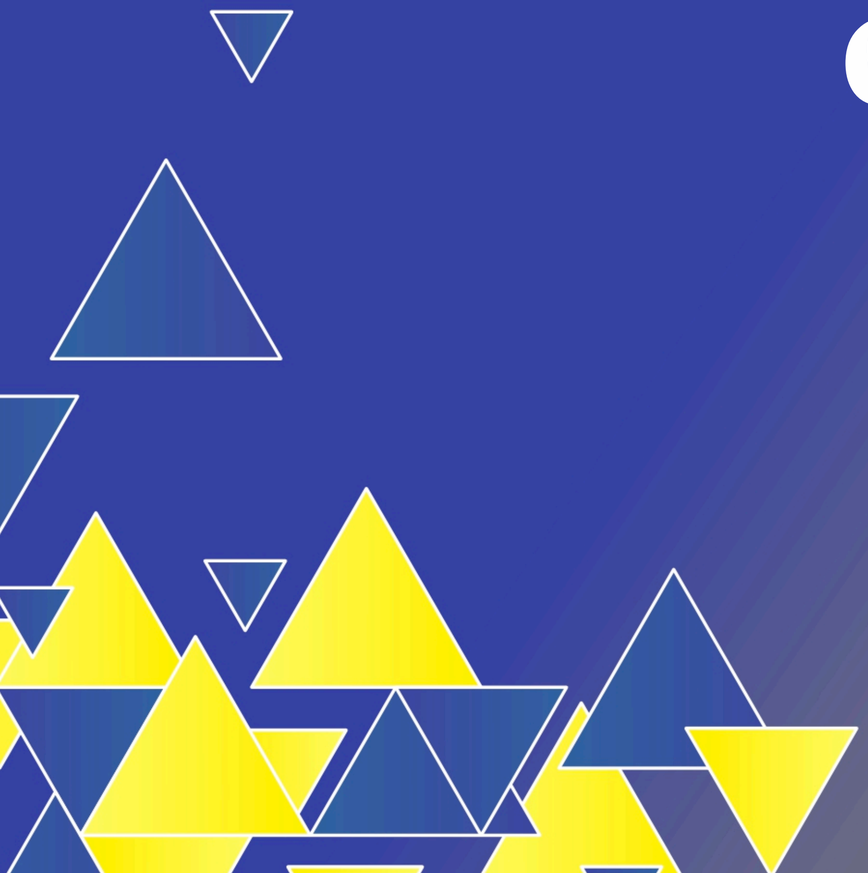
Python
fwdays

The art of data engineering

Andrii Soldatenko

TVTime



The art of Data engineering

14 December 2019,
Remote :)

@a_soldatenko

Andrii Soldatenko

- Senior Data Engineer TVTime
- Apache Airflow contributor
- Public Speaker at many conferences, committee member of different PyCons
- blogger at https://asoldatenko.com

@a_soldatenko

Registered
Users

Episodes Tracked
Across 90k+ Shows

Daily Active
Users

Years of
Tracking Data

Countries
Represented

Platforms
Covered

@a_soldatenko

“…The Web was done by amateurs” 🎉

- Alan Kay

@a_soldatenko

TL;DR

Mat Velloso
@matvelloso

Follow

Difference between machine learning and AI:

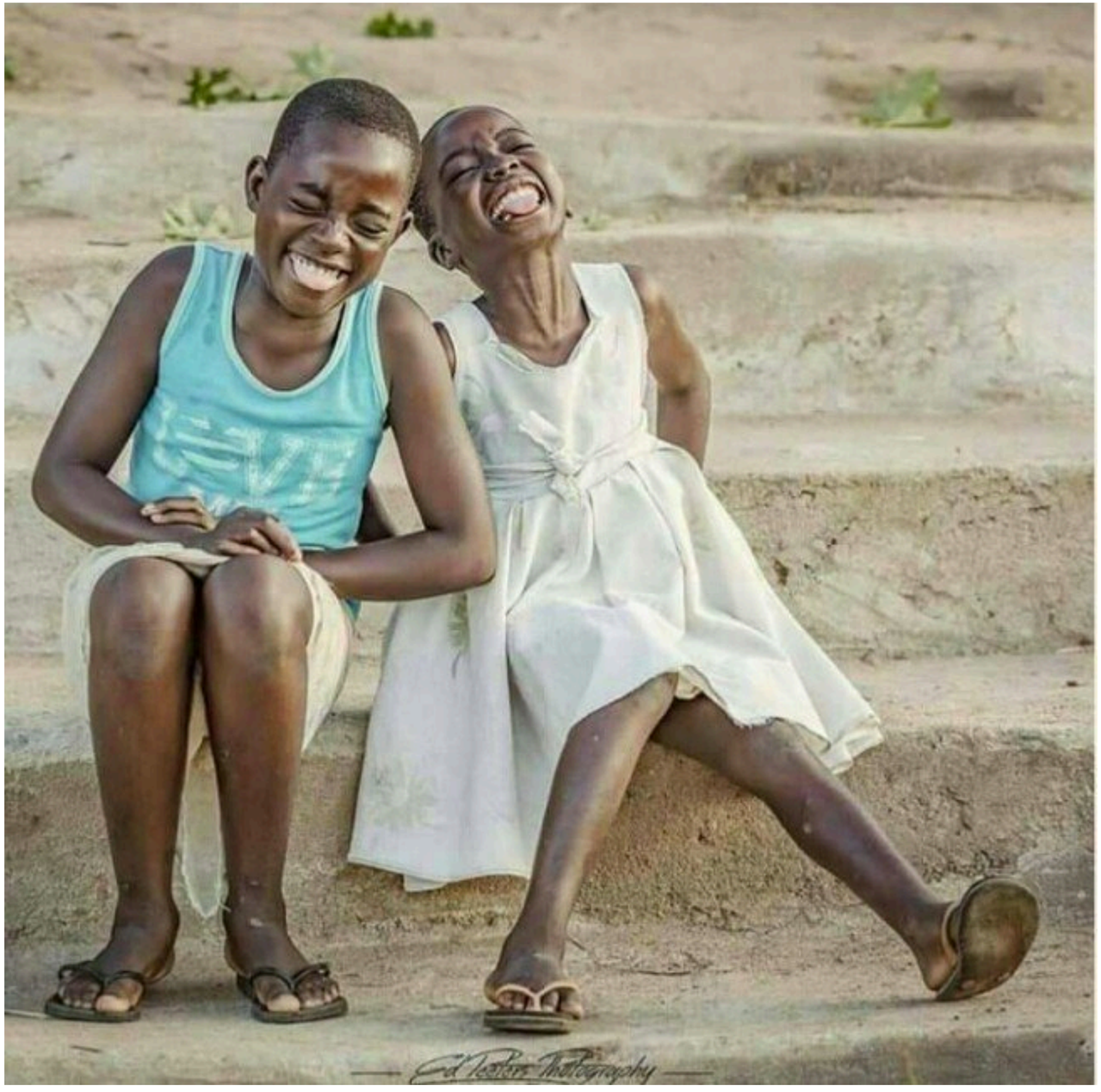
If it is written in Python, it's probably machine learning

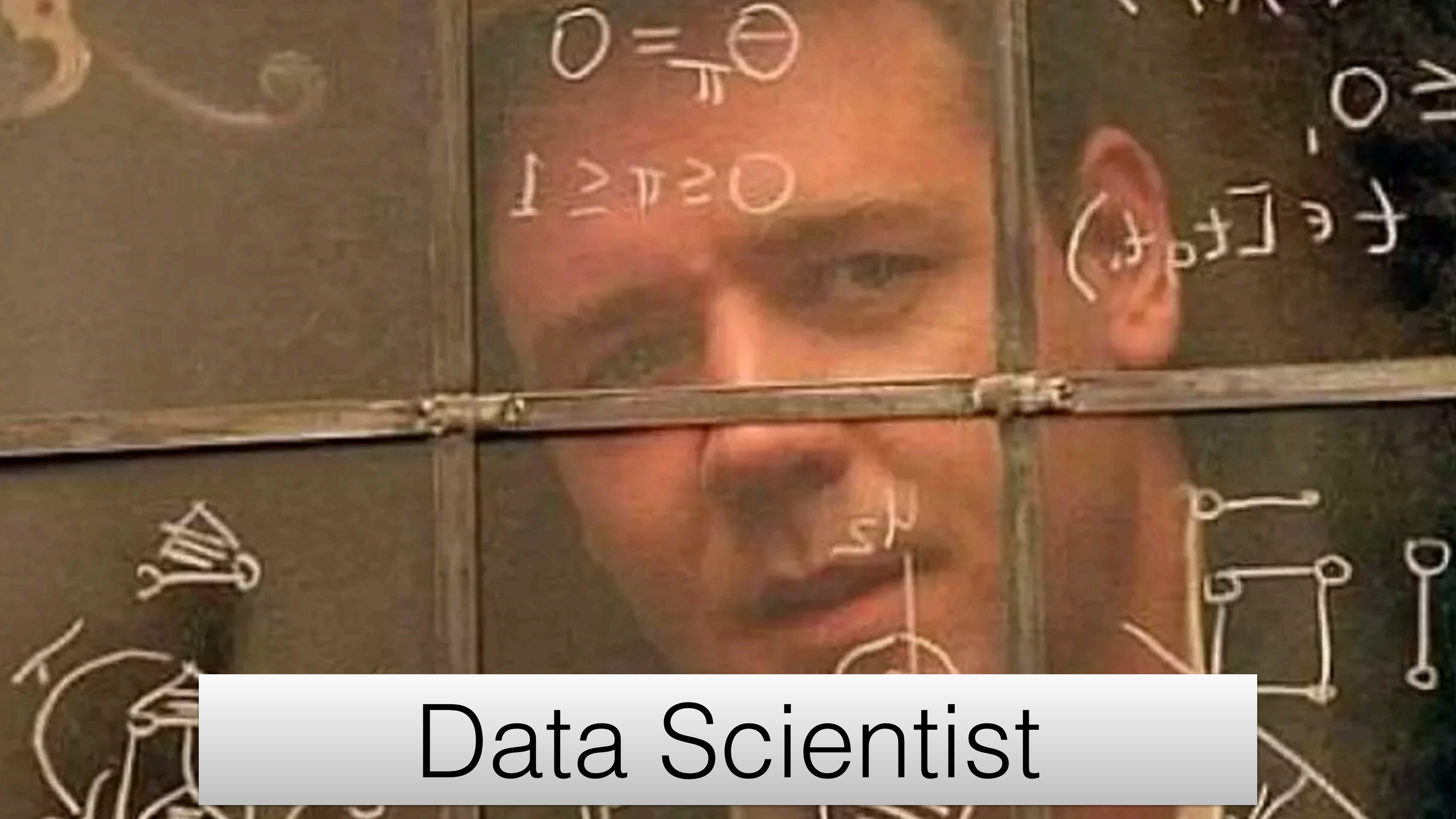
If it is written in PowerPoint, it's probably AI

5:25 PM - 22 Nov 2018

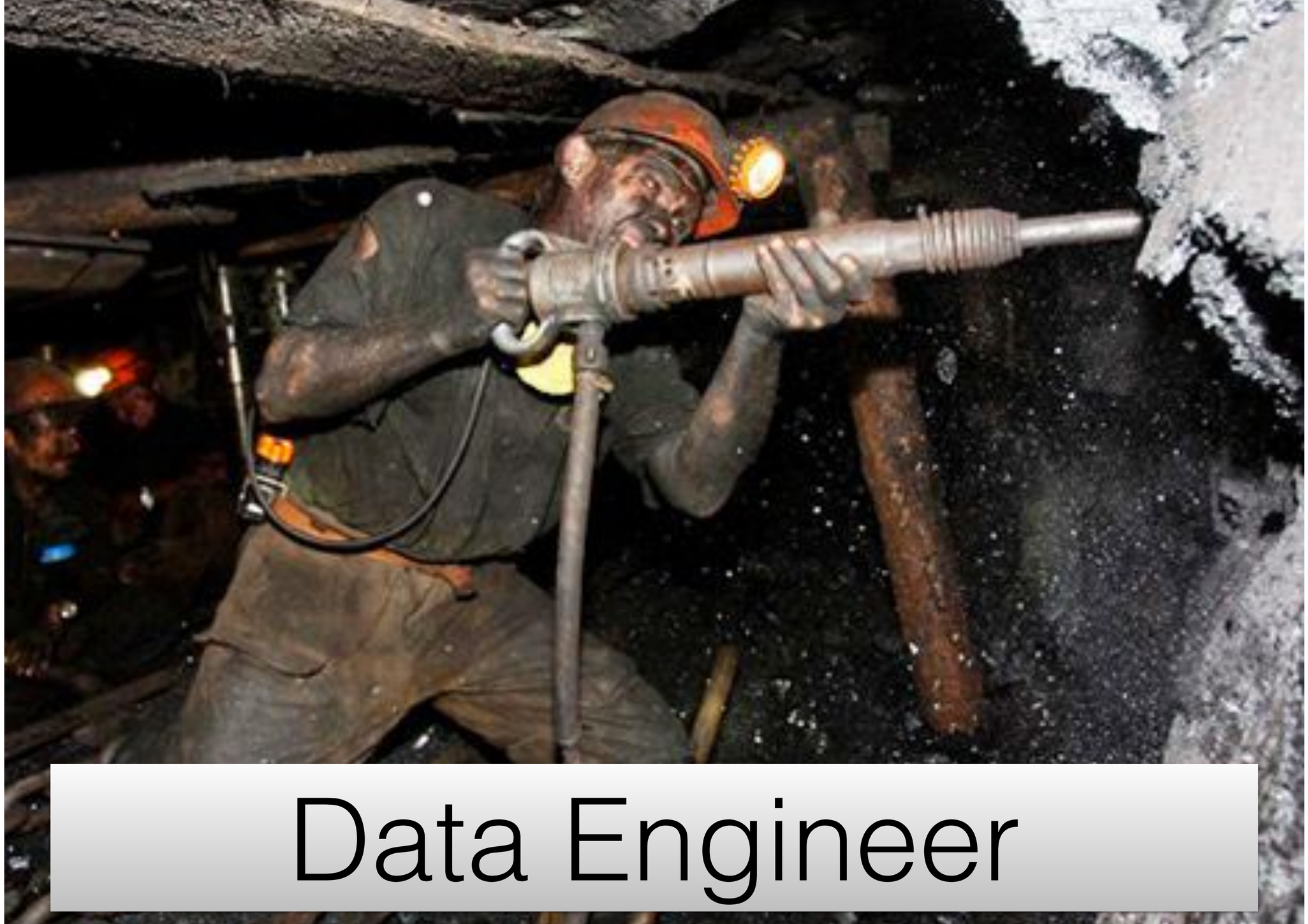
1,186 Retweets **3,333** Likes

41 1.2K 3.3K



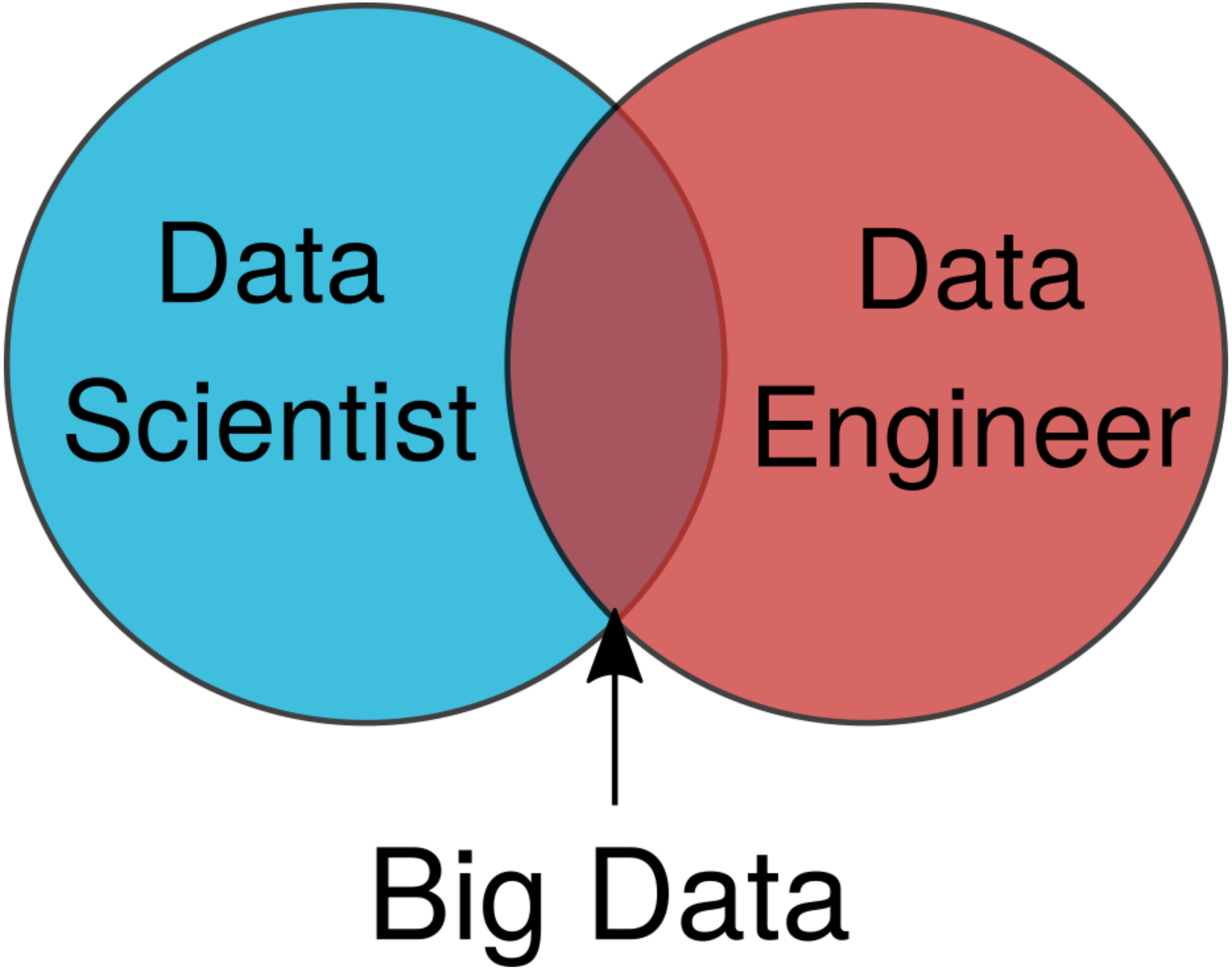


Data Scientist



Data Engineer

Data Engineering:
A quick definition

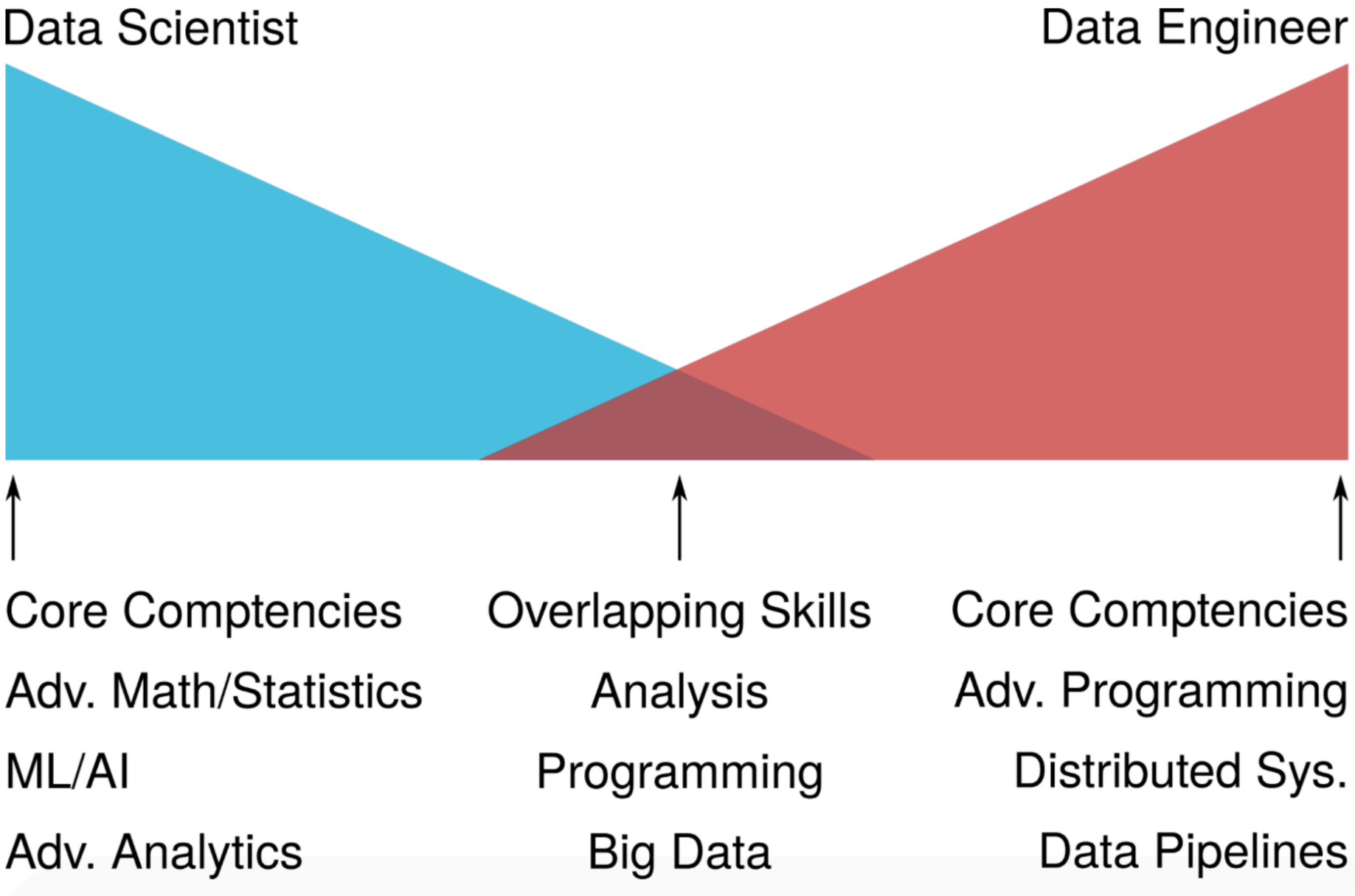

Data
Scientist
Data
Engineer
Big Data

Twitter: 600 million tweets per day.

Facebook: 600 terabytes of incoming data each day, from 1.6 billion active users.

Google: 3.5 billion search queries per day.

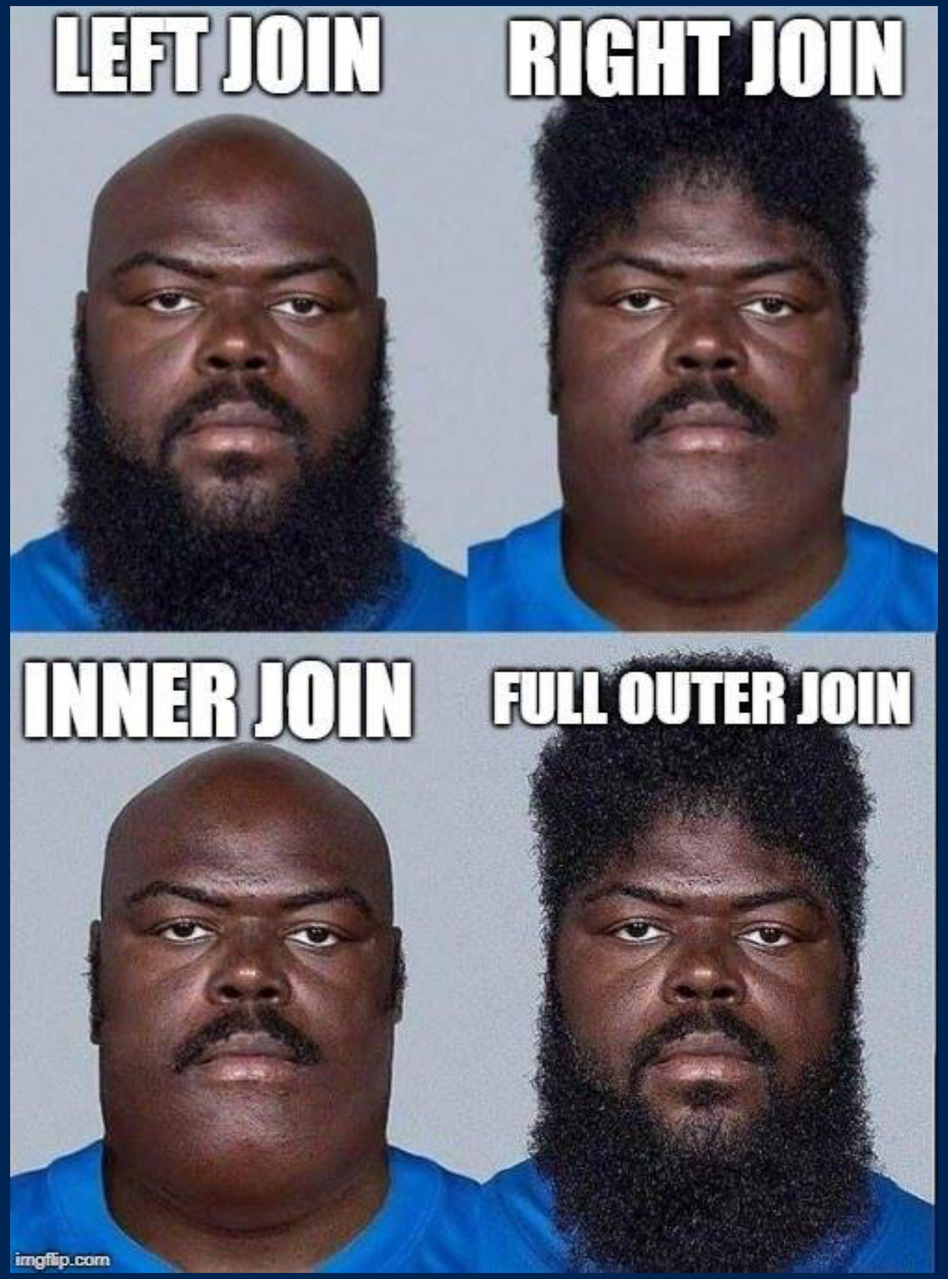
Instagram: 52 million new photos per day


Data Scientist
Data Engineer
Core Comptencies
Adv. Math/Statistics
ML/AI
Adv. Analytics
Overlapping Skills
Analysis
Programming
Big Data
Core Comptencies
Adv. Programming
Distributed Sys.
Data Pipelines


What are data
engineers focused on?

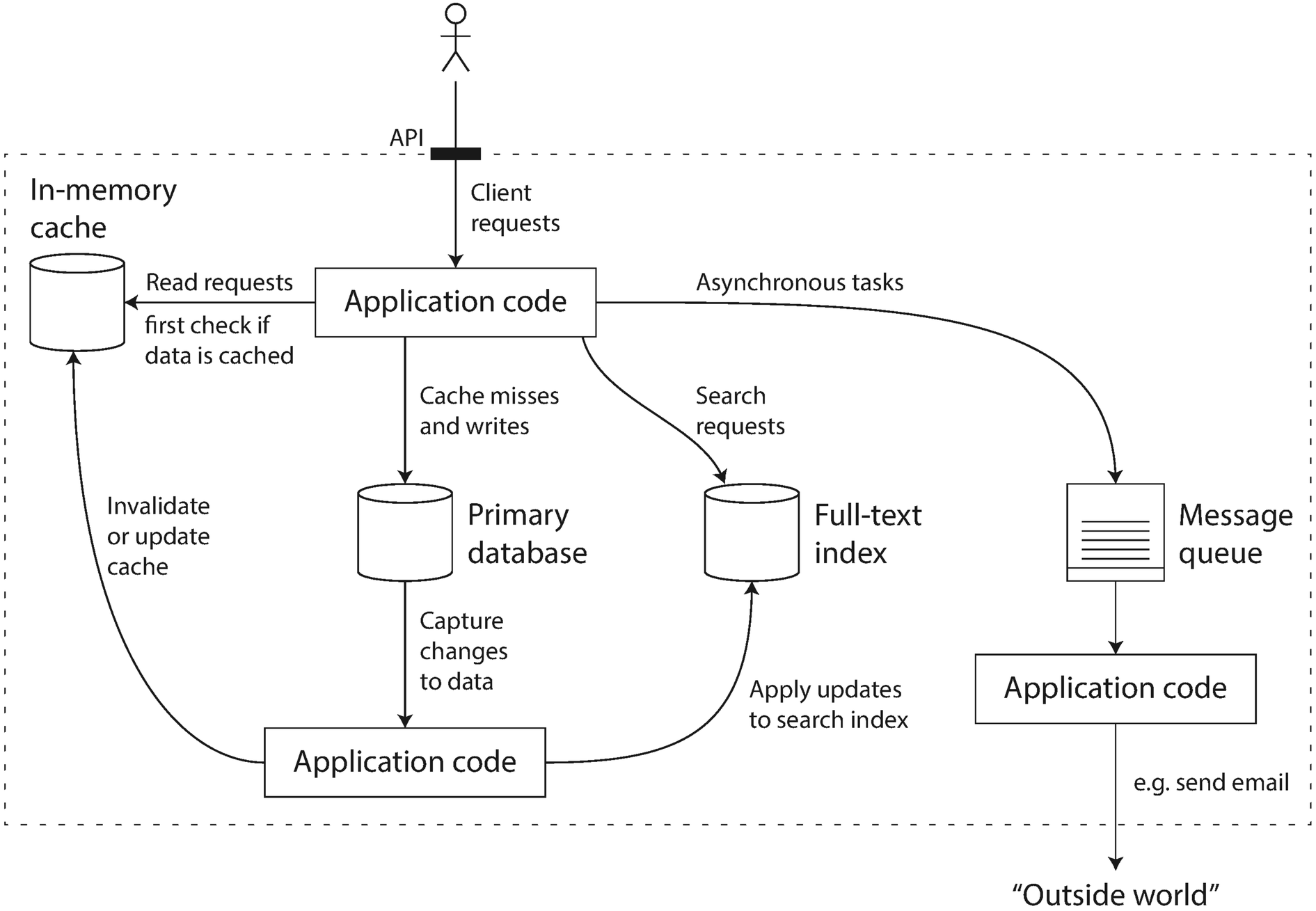
How to Become a Data Engineer? 👀

What skills do data engineers need?


LEFT JOIN
RIGHT JOIN
INNER JOIN
FULL OUTER JOIN
imgflip.com


Atlanta Linux Showcase
February 9-11, 2001

Possible architecture for a data system


API
In-memory cache
Client requests
Read requests
first check if data is cached
Application code
Asynchronous tasks
Cache misses and writes
Search requests
Invalidate or update cache
Primary database
Full-text index
Message queue
Capture changes to data
Apply updates to search index
Application code
Application code
e.g. send email
"Outside world"

Cron + bash script 🔥

```
#!/bin/bash
```

```
~root: env X="() { :;} ; echo shellshock" /bin/sh -c "echo completed"

> shellshock
> completed
```

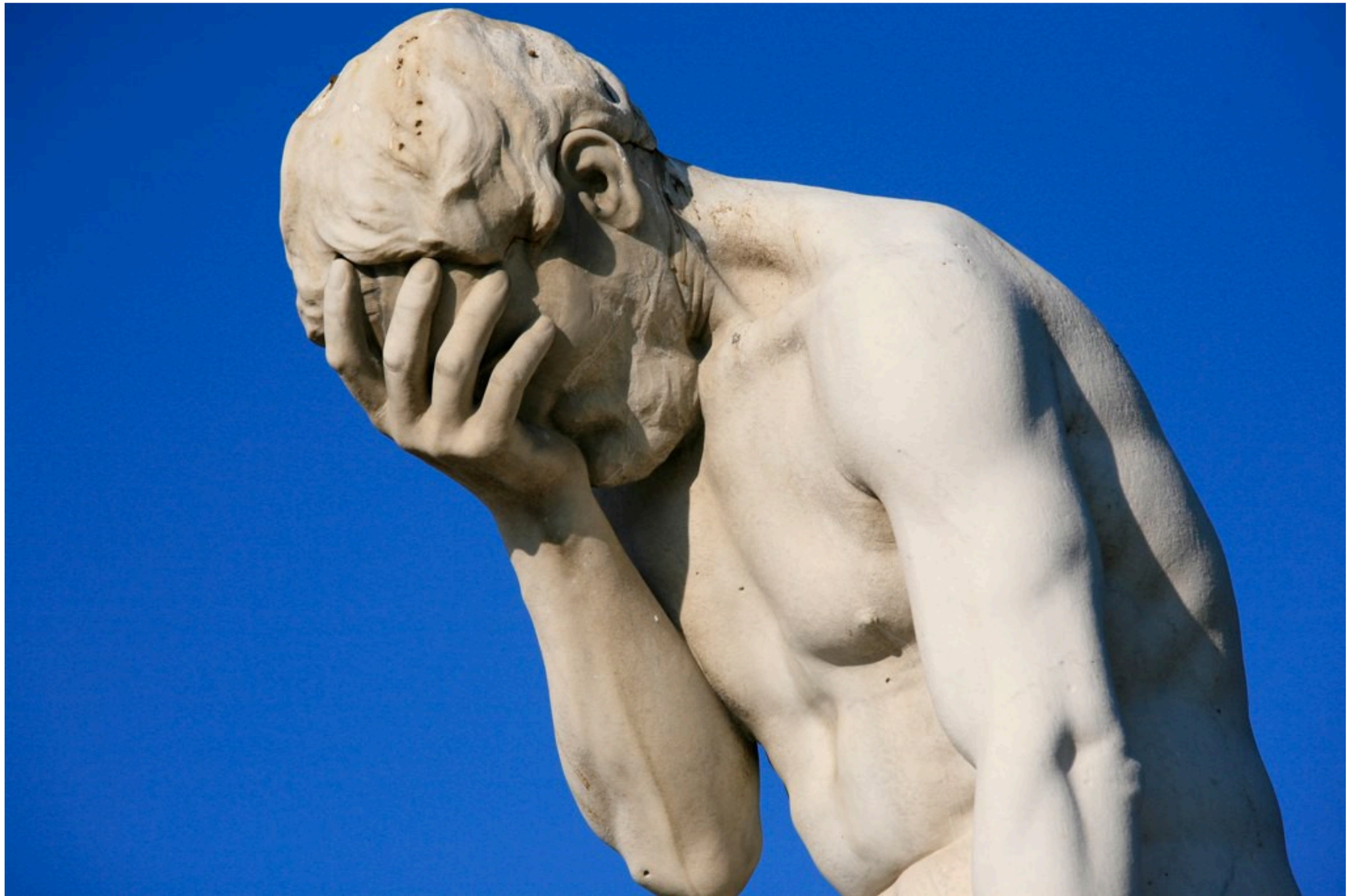
```
tes after the hour (0-59)
hour format (0-23).
ay of the month (1-31)
#  |    |    |    |_____________ 4. Month - Month of the year (1-12)
#  |    |    |    |
#  |    |    |    |_____________5. Weekday - Day of the week. (0-6, where 0 indicates Sun$
#  |    |    |    |    |
#__|____|____|____|____|__Command_____________________________________________________
# Web stats at https://[redacted]
   1   */1     *     *     *    perl /usr/lib/cgi-bin/awstats.pl -config=web -update >/dev/null
   1   */1     *     *     *    perl /usr/lib/cgi-bin/awstats.pl -config=smtp -update >/dev/null
# Backup LDAP at 03:00
   0     3     *     *     *    bash /var/vmail/backup/backup_openldap.sh
# Backup mysql at 03:30
  30     3     *     *     *    bash /var/vmail/backup/backup_mysql.sh
```

Simple data pipeline 🔥

```
TL;DR
0 * * * * my_bash_pipeline.sh


#!/bin/bash
psql -U user -d database_name -c \
"COPY country FROM '/tmp/
country_data';"
```

I’m sorry Cron, I’ve met Airflow :)



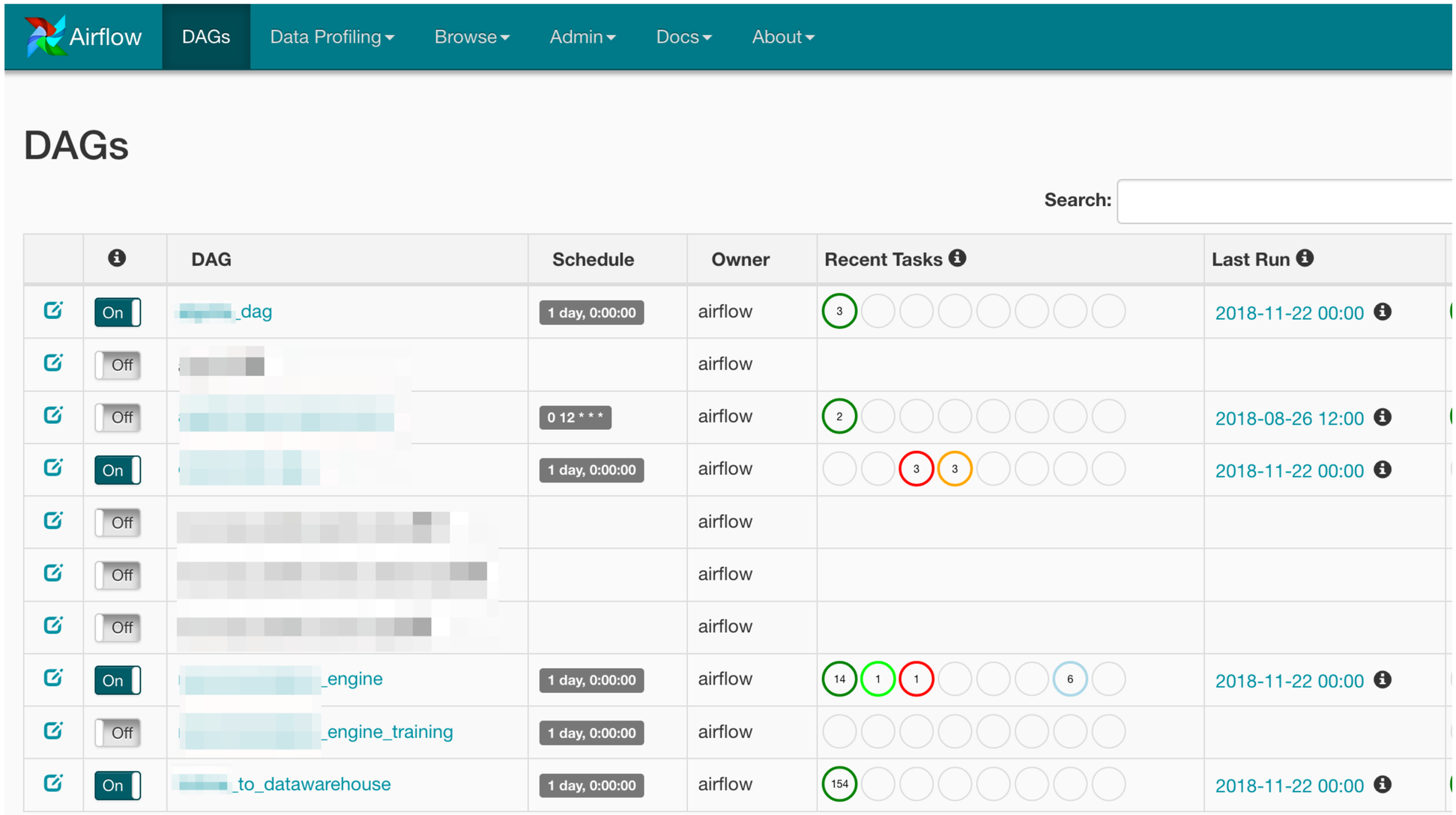
Data Pipelins:
Open source TooIZZZ

OS workflow tools:

	Luigi	Airflow	Pinball
repo	https://github.com/spotify/luigi	https://github.com/airbnb/airflow	https://github.com/pinterest/pinball
docs	http://luigi.readthedocs.org	https://airflow.readthedocs.org	none
github forks	1786	3494	58
github stars	10520	10072	922
github watchers	510	626	126
commits in last 30 days	3700 commits	5618 commits	133 commits
architecture			
web dashboard	not really, minimal	very nice	yes
code/dsl	code	code	python dict + python code
files/datasets	yes, targets	not really, as special tasks	?
calendar scheduling	no, use cron	yes, LocalScheduler	yes
backfill jobs	yes	yes	?
persists state	kindof	yes, to db	yes, to db
tracks history	yes	yes, in db	yes, in db
code shipping	no	yes, pickle	workflow is shipped using pickle, jobs are not?
priorities	yes	yes	?
parallelism	yes, workers, threads per workers	yes, workers	?
control parallelism	yes, resources	yes, pools	?
cross-dag deps	yes, using targets	yes, using sensors	yes
finds new deployed tasks	no	yes	?
executes dag	no, have to create special sink task	yes	yes
multiple dags	no, just one	yes, also several dag instances (dagruns)	yes
scheduler/workers			

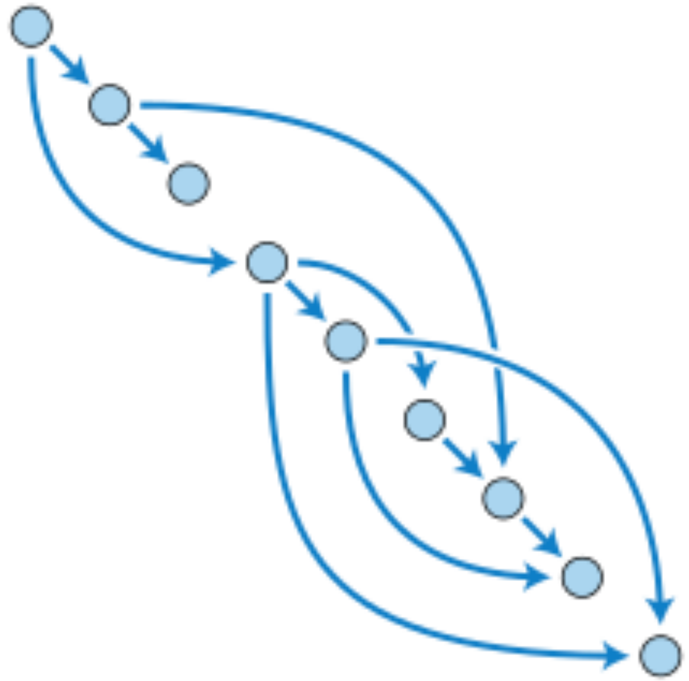
https://docs.google.com/spreadsheets/d/1wfKibn4yS7AshPzY8ch2Dq_VmQVKnGSXGj08A2u1YQI/edit#gid=0

Airflow

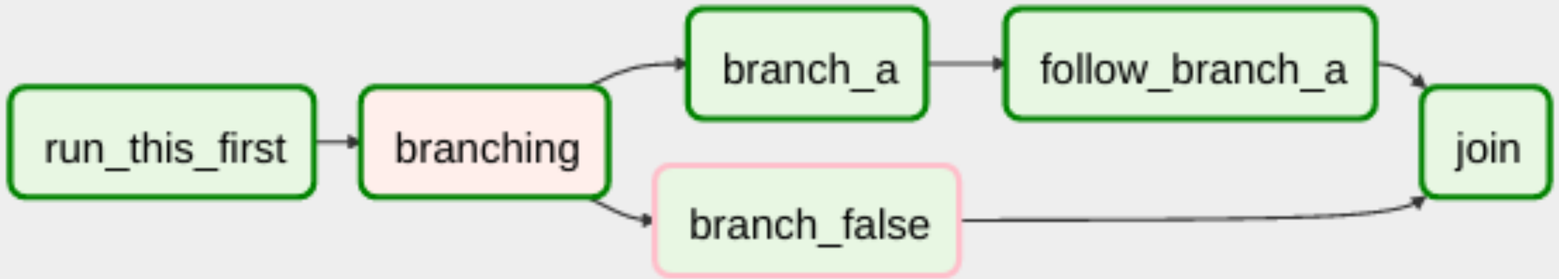

Airflow
DAGs
Data Profiling
Browse
Admin
Docs
About
DAGs
Search:
DAG
Schedule
Owner
Recent Tasks
Last Run
On
_dag
1 day, 0:00:00
airflow
2018-11-22 00:00
Off
airflow
Off
0 12 * * *
airflow
2018-08-26 12:00
On
1 day, 0:00:00
airflow
2018-11-22 00:00
Off
airflow
Off
airflow
Off
airflow
On
_engine
1 day, 0:00:00
airflow
2018-11-22 00:00
Off
_engine_training
1 day, 0:00:00
airflow
On
_to_datawarehouse
1 day, 0:00:00
airflow
2018-11-22 00:00

Airflow Concepts

Airflow Core Ideas: DAGS

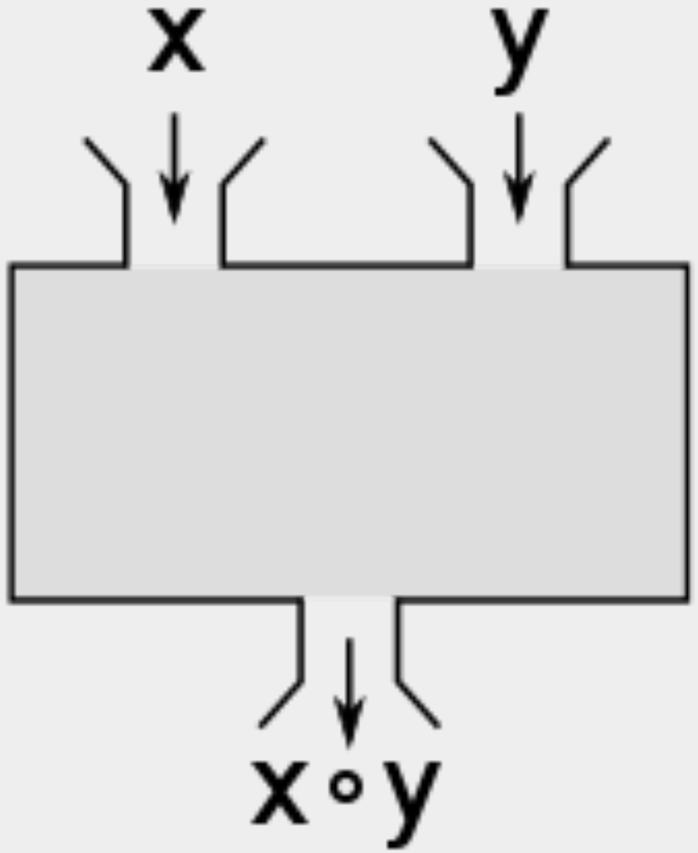


Airflow Core Ideas



Airflow Core Ideas: Operators 👩‍🚀

- BashOperator
- PythonOperator
- MySqlOperator
- S3FileTransformOperator … you get the idea



https://airflow.apache.org/concepts.html

Return to real world

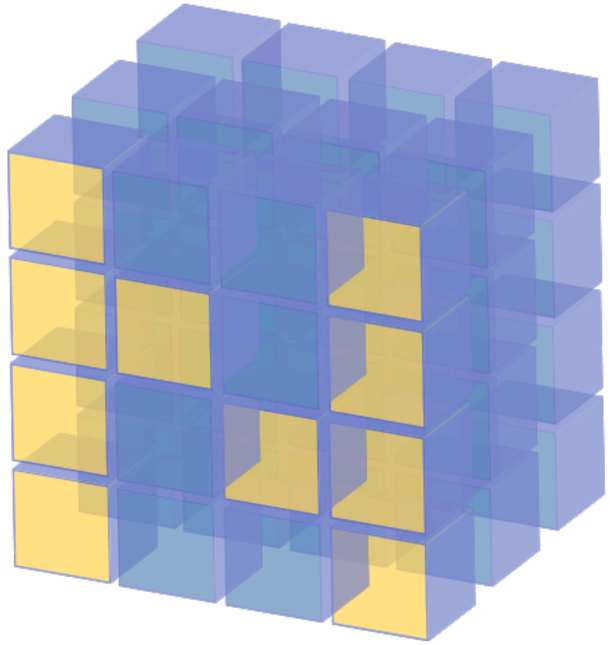
ML Dependency hell:

SciPy

spaCy

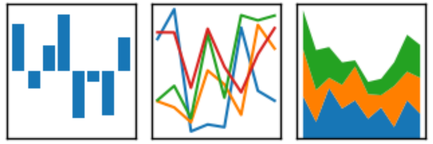
scikit learn

python™
Natural Language Analyses
with NLTK



pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



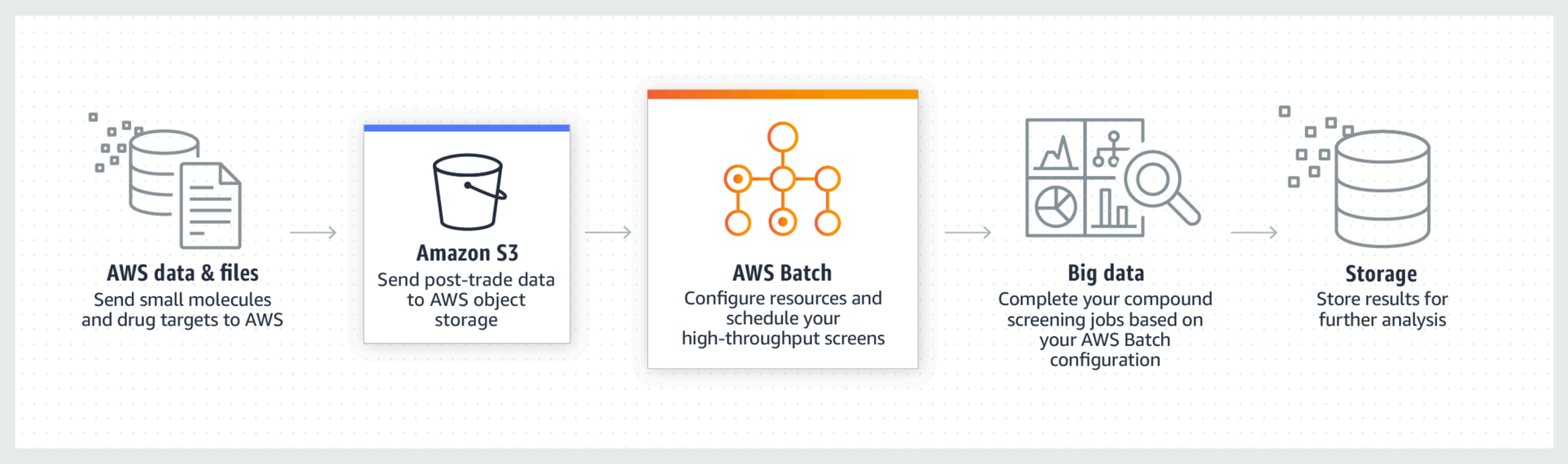
NumPy

TensorFlow

AWS: the good, the bad and the ugly

ML Dependency hell:

Case #1 Amazon Batch:



AWS Batch and Airflow

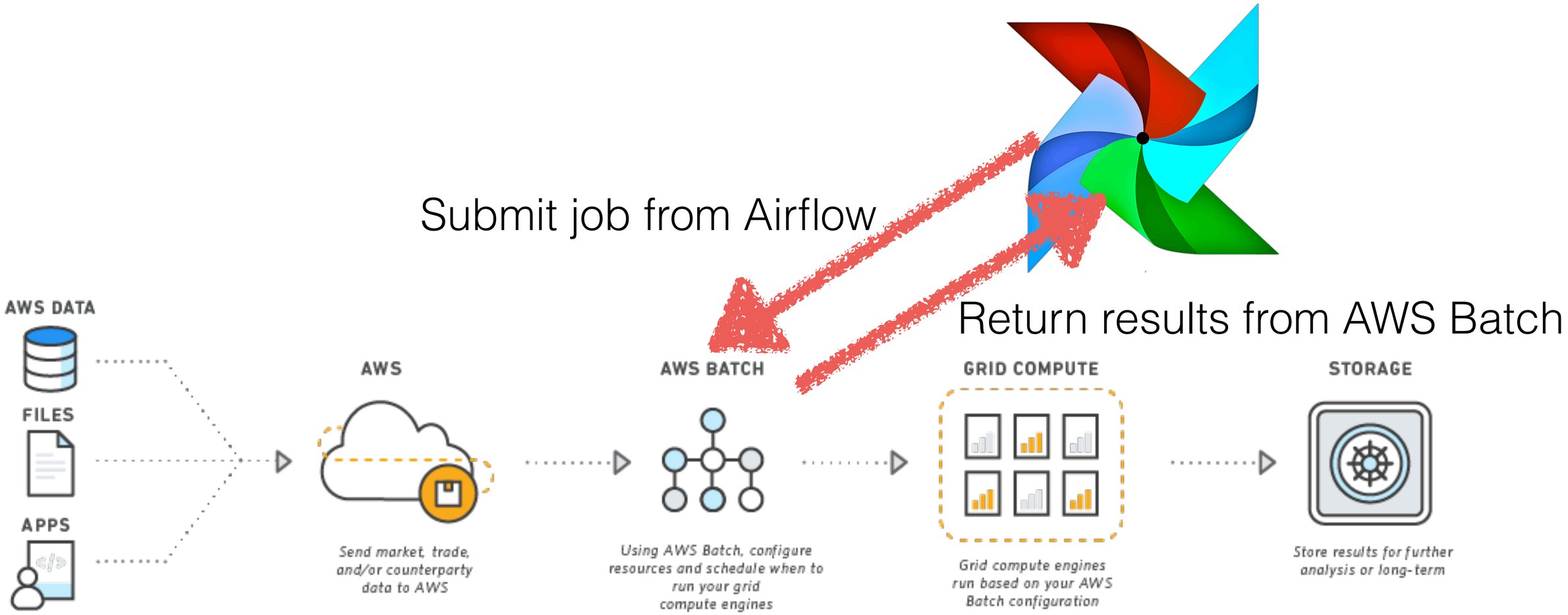
AWS Batch and Airflow

AWS Batch and Airflow: Example

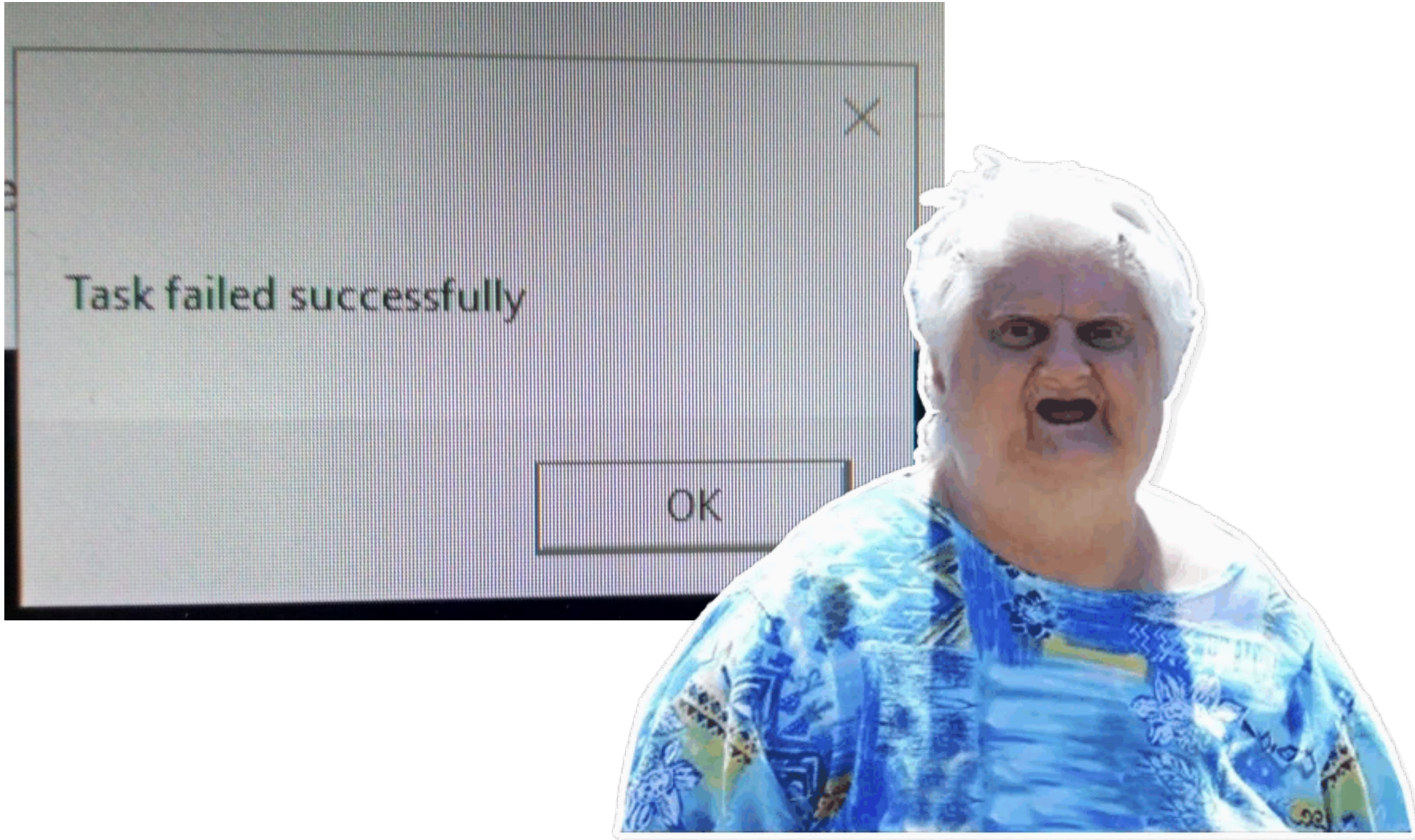
```
FROM amazonlinux:latest
RUN yum -y groupinstall 'Development Tools'
...
WORKDIR /scratch
RUN git clone https://github.com/cjlin1/
libmf.git /tmp/libmf && cd /tmp/libmf && make
ENTRYPOINT ["/usr/local/bin/entrypoint.sh"]
```

LIBMF is a library for large-scale sparse matrix factorization. For the optimization problem it solves and the overall framework, please refer to [3].

AWS Batch and Airflow



AWS Batch and Airflow



22G volume mounted by default , ~ 10Gb

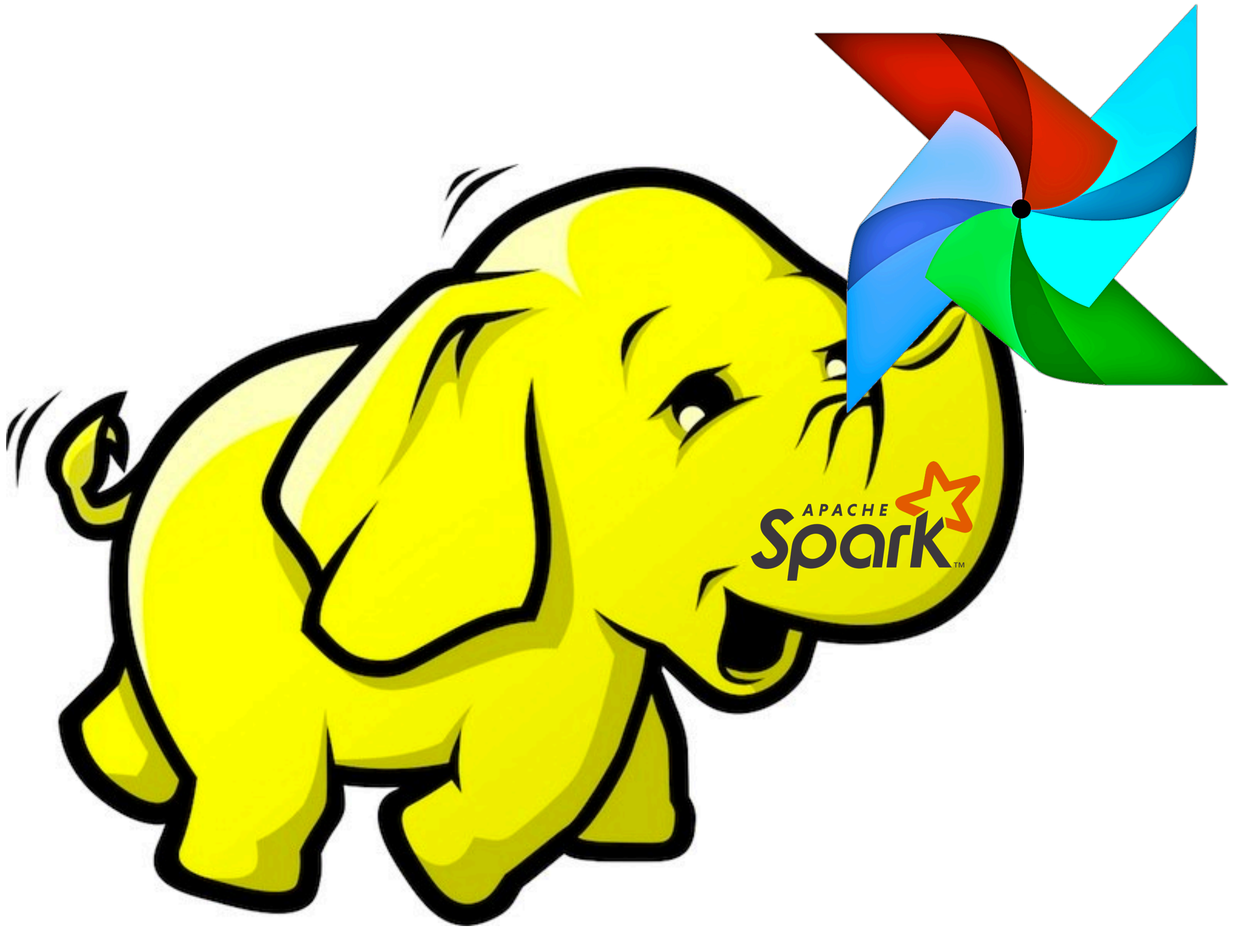
https://forums.aws.amazon.com/thread.jspa?threadID=250705

Solution =)

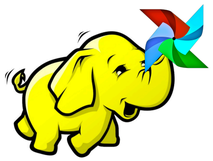
- You need to create EC2 based on ECS optimized AMIs
- Attach another volume separate from your root volume, you will want to modify the /etc/fstab
- Create image (AMI) Or use (ami-XXXXX with extra 20Gb space)
- Create AWS Batch compute environment based on AMI

- useful link https://aws.amazon.com/blogs/compute/building-high-throughput-genomic-batch-workflows-on-aws-batch-layer-part-3-of-4/)

Case #2 EMR Spark and Airflow

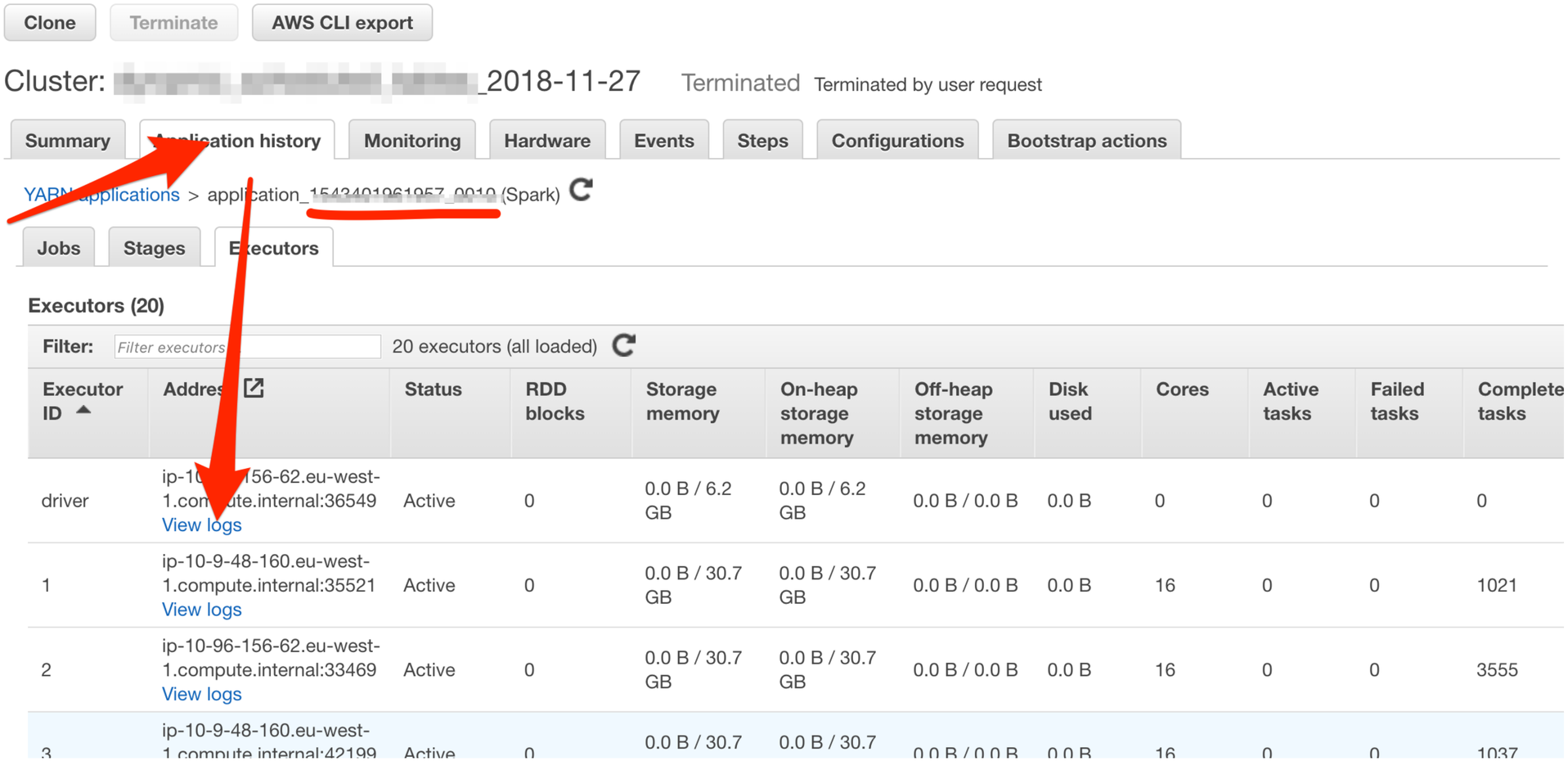


EMR and Airflow



airflow/contrib/operators/**emr_add_steps_operator.py**
airflow/contrib/operators/**emr_create_job_flow_operator.py**
airflow/contrib/operators/**emr_terminate_job_flow_operator.py**

Debug EMR cluster:



Clone | Terminate | AWS CLI export

Cluster: [redacted]_2018-11-27 Terminated Terminated by user request

Summary | Application history | Monitoring | Hardware | Events | Steps | Configurations | Bootstrap actions

YARN applications > application_1543401961957_0010 (Spark)

Jobs | Stages | Executors

Executors (20)

Filter: Filter executors 20 executors (all loaded)

Executor ID	Address	Status	RDD blocks	Storage memory	On-heap storage memory	Off-heap storage memory	Disk used	Cores	Active tasks	Failed tasks	Complete tasks
driver	ip-10-96-156-62.eu-west-1.compute.internal:36549 View logs	Active	0	0.0 B / 6.2 GB	0.0 B / 6.2 GB	0.0 B / 0.0 B	0.0 B	0	0	0	0
1	ip-10-9-48-160.eu-west-1.compute.internal:35521 View logs	Active	0	0.0 B / 30.7 GB	0.0 B / 30.7 GB	0.0 B / 0.0 B	0.0 B	16	0	0	1021
2	ip-10-96-156-62.eu-west-1.compute.internal:33469 View logs	Active	0	0.0 B / 30.7 GB	0.0 B / 30.7 GB	0.0 B / 0.0 B	0.0 B	16	0	0	3555
3	ip-10-9-48-160.eu-west-1.compute.internal:42199	Active	0	0.0 B / 30.7	0.0 B / 30.7	0.0 B / 0.0 B	0.0 B	16	0	0	1037

As a final step
load data to warehouse service:

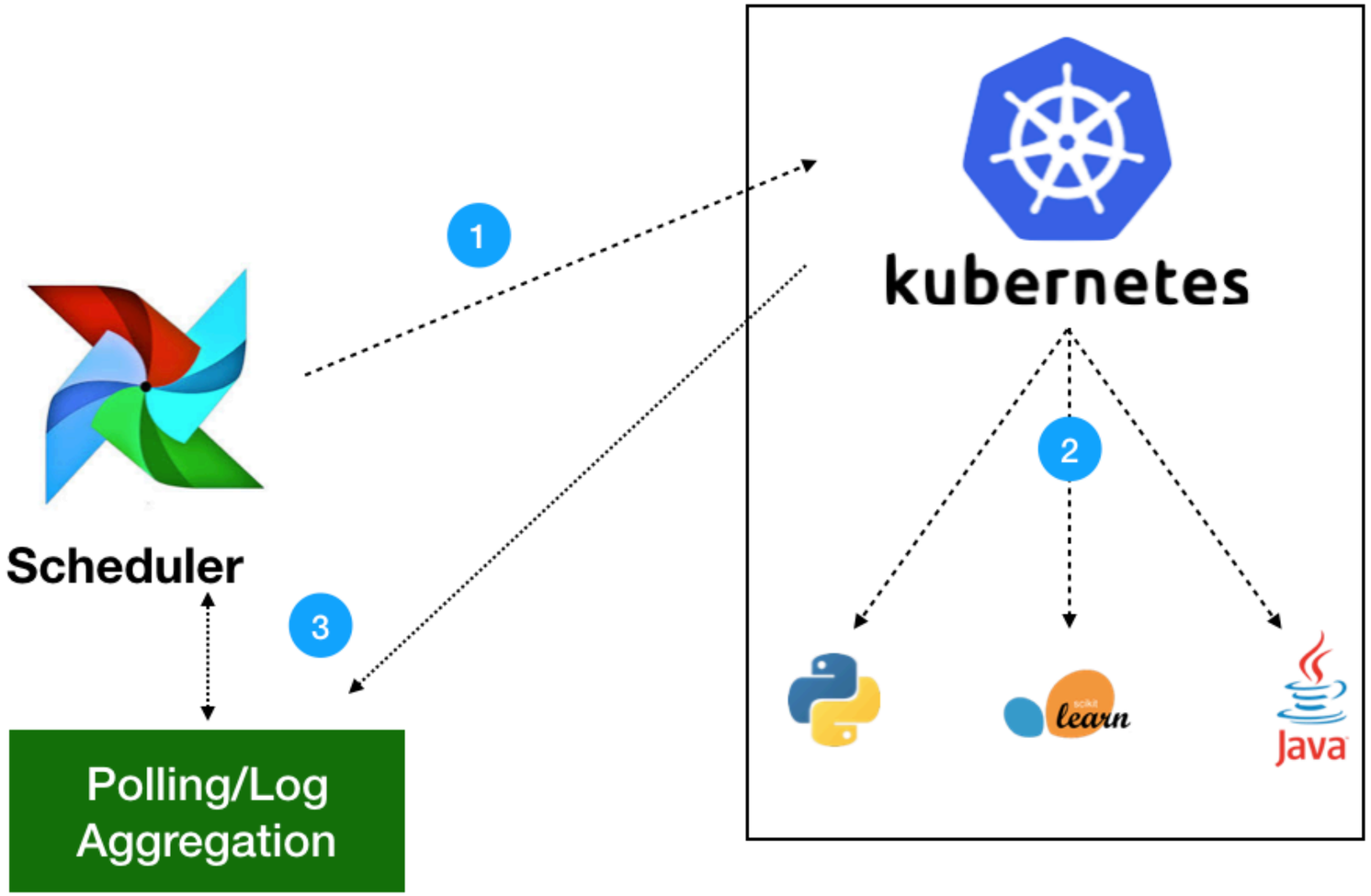
Amazon **Redshift**

As a final step
load data to warehouse service:


```
COPY INTO table (colA, colB)
    from (SELECT (colA, colB)
    FROM @s3_stage_parquet)
    ON_ERROR = CONTINUE
```

snowflake®

Case #3 Kubernetes Pod Executor and Airflow



CI/CD pipelines

CI/CD pipelines

- unit tests/integration tests 🍸
- code quality checks 🔫
- automatic deployment after merge into master 🌀
- Pull Request code review flow 🍒

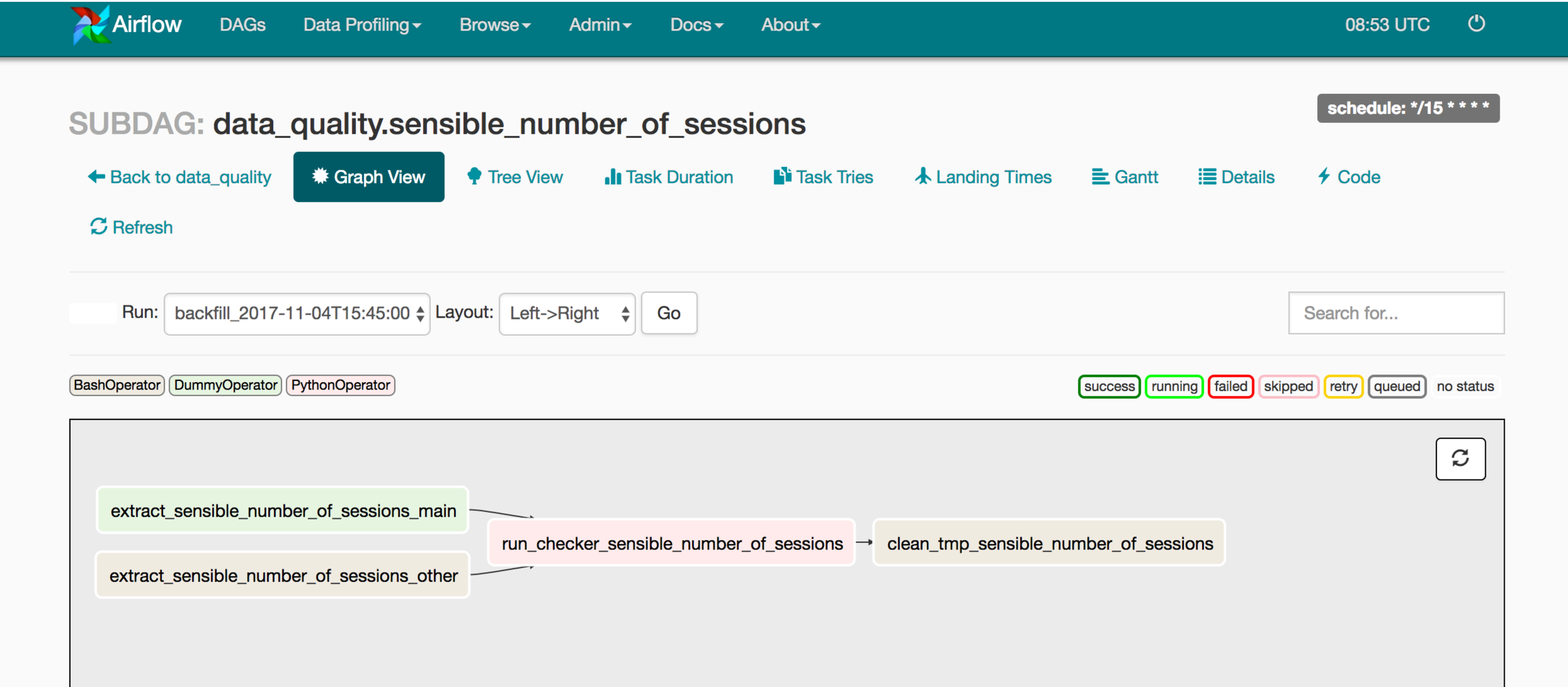
unit tests/integration tests 🍸

```python
@pytest.mark.parametrize('dag_file', DAG_FILES)
def test_import_dag_files(dag_file):
    """Import dag files and check for DAG."""
    module_name, _ = os.path.splitext(dag_file)
    module_path = os.path.join(DAG_PATH, dag_file)
    mod_spec = importlib.util.spec_from_file_location(module_name, module_path)
    module = importlib.util.module_from_spec(mod_spec)
    mod_spec.loader.exec_module(module)
    assert any(
        isinstance(var, af_models.DAG)
        for var in vars(module).values())
```

Data quality checkers

```
COPY DATA to NEW_TABLE
APPLY DATA QUALITY CHECKS IF ERROR -> SEND ALERTS EXIT(1)
BACKUP ORIGINAL TABLE
RENAME NEW_TABLE to ORIGINAL_TABLE
```

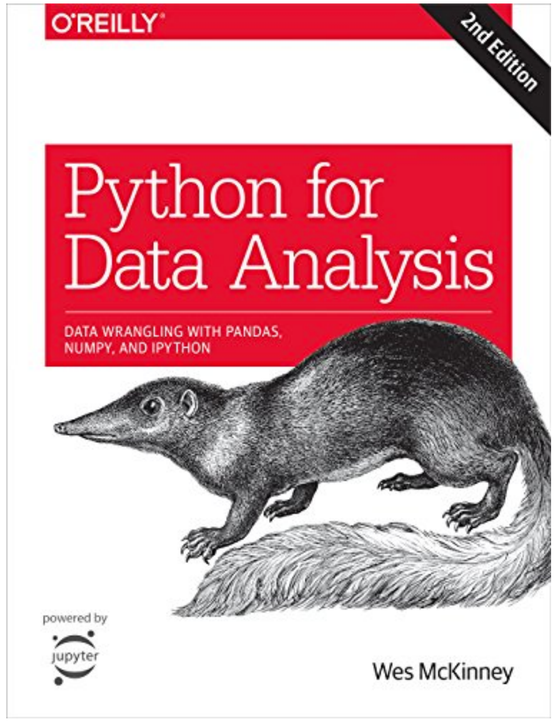
Data quality checkers

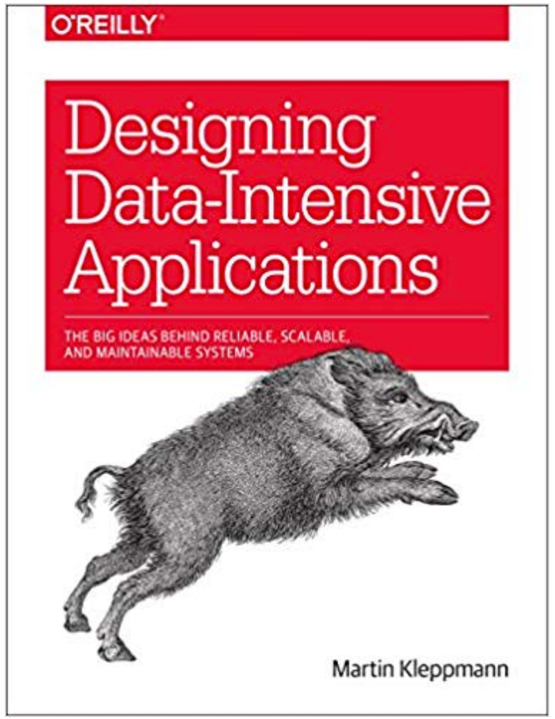


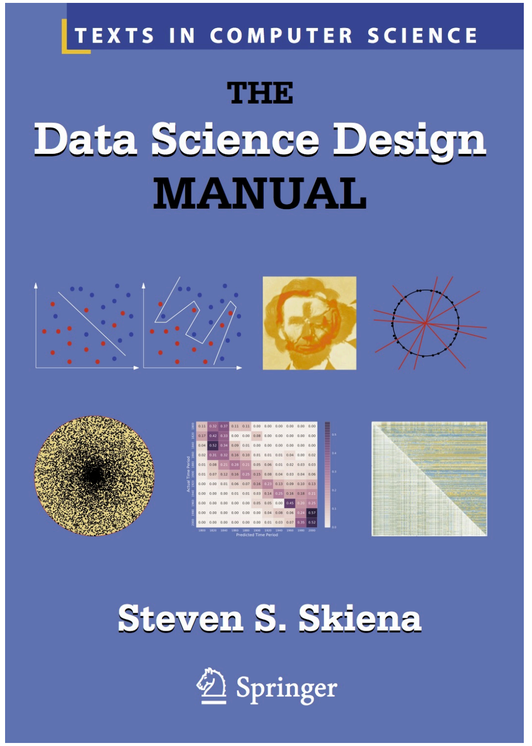
Learn more

Learn more

- Awesome Data Engineering (igorbarinov/awesome-data-engineering)
- Awesome Apache airflow (jghoman/awesome-apache-airflow)







Conclusion:

- Data engineering new trend with old roots 🌴
- Data intensive apps is fun 📈
- ETL code base is just yet another source code 🛠️
- You must know better than others you data storages بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Thank You

Thank You

andrii.soldatenko@gmail.com

@a_soldatenko

Questions

@a_soldatenko