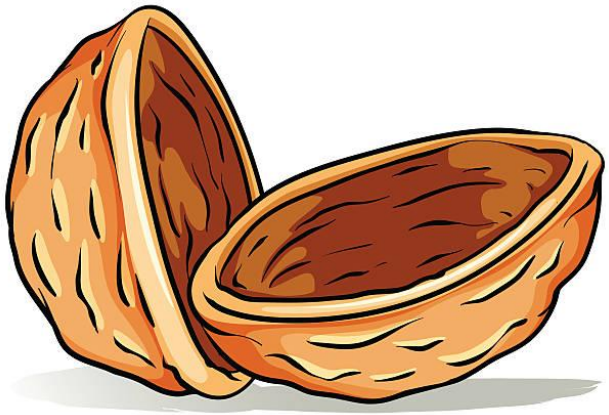
Cron em esteróides:

Apache Airflow (Brain-on Hands-on)

Olá, nózes!



Nitai Bezerra

nitai.silva
@planejamento.gov.br

t. 2020-1551

@nitais

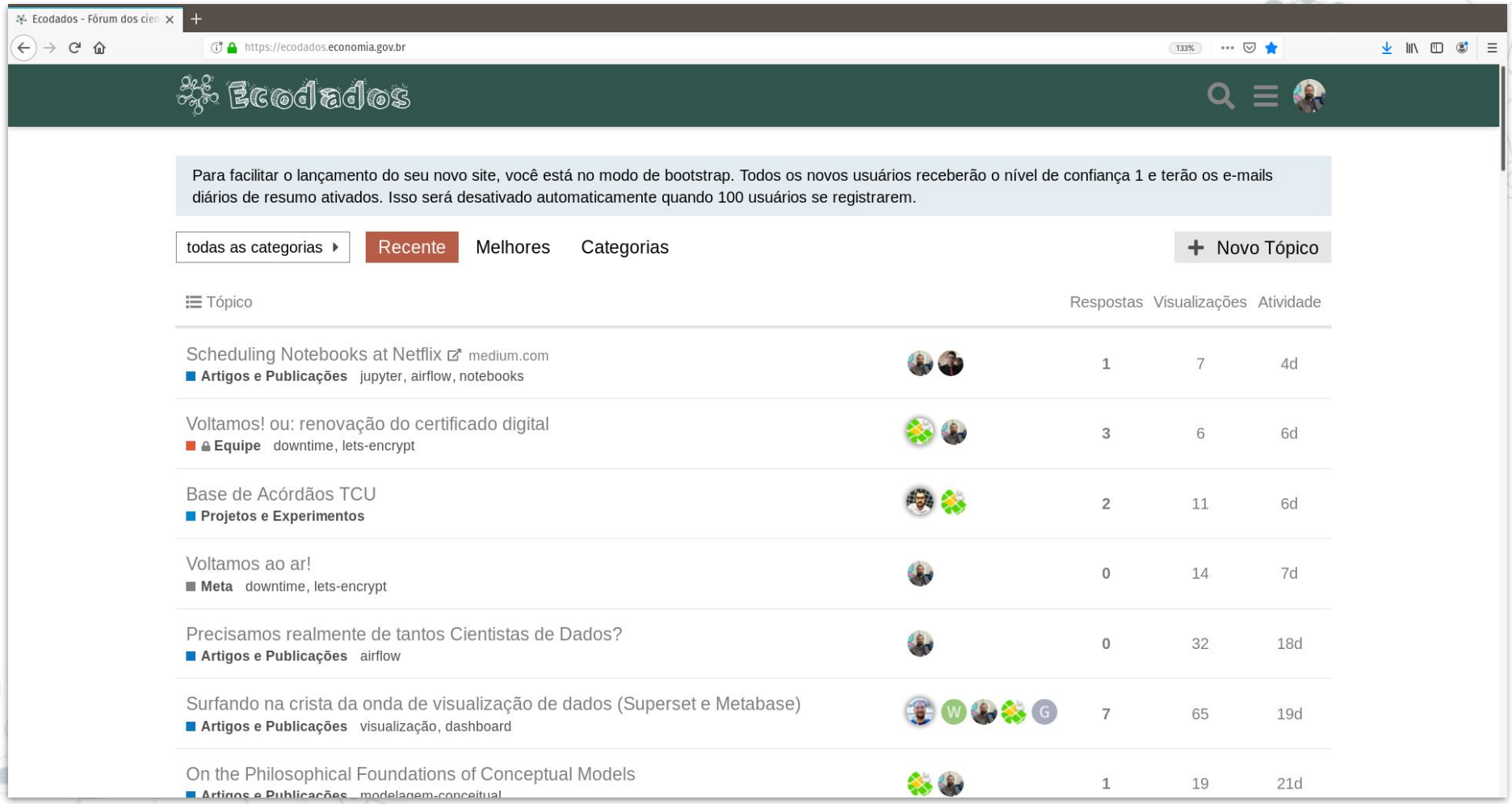
Vitor Bellini

vitor.bellini
@planejamento.gov.br

t. 2020-1504

@podefalar

ecodados.economia.gov.br



E aí!? Como vai ser?

1. Quebrando o gelo gelado
2. Ao infinito e além: filosofia e arquitetura Airflow
3. Airflow UI, seu lindo!
4. DAG: que danado é isso!?
5. Baleia intro (Docker)
6. Botar pra rodar
7. SELECT test FROM oficina
8. [break] Ufa, DESCANSO (20 min.)

E aí!? Como vai ser?

8. [break] Ufa, DESCANSO
9. Hello World: Seu primeiro robô (ou não)
10. Baby Steps: Implementação de uma DAG pra chamar de sua
11. Exemplos DAGs ME
12. Dicas e Truques (quem souber me conta)
13. Feedback

1.

Quebrando o gelo gelado

Origami + Enigma

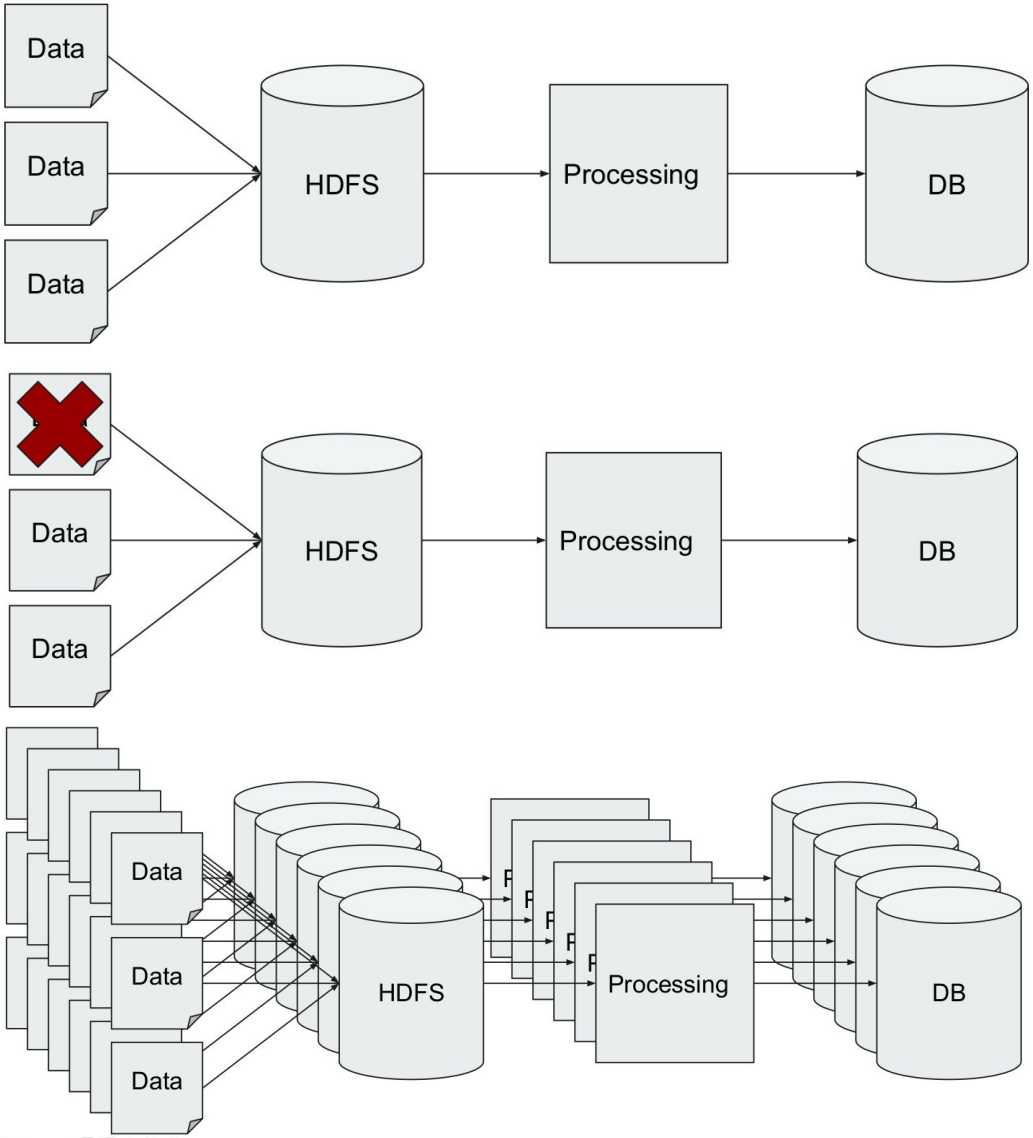
Washington Oliveira

Porque estamos aqui!?

2.

Ao infinito e além: filosofia e arquitetura Airflow

Porque usar?


Data
Data
Data
HDFS
Processing
DB
Data
Data
HDFS
Processing
DB
Data
Data
Data
HDFS
Processing
DB

Para isso e mais...

- Substituir rotinas Cron
- Tolerancia à falhas
- Regras de dependência
- Python!!!
- Versionamento dos códigos

Para isso e mais...

- Tratamento de erros
- Relatórios e alertas em falhas
- Monitoramento de pipelines
- UI bonita pacas!

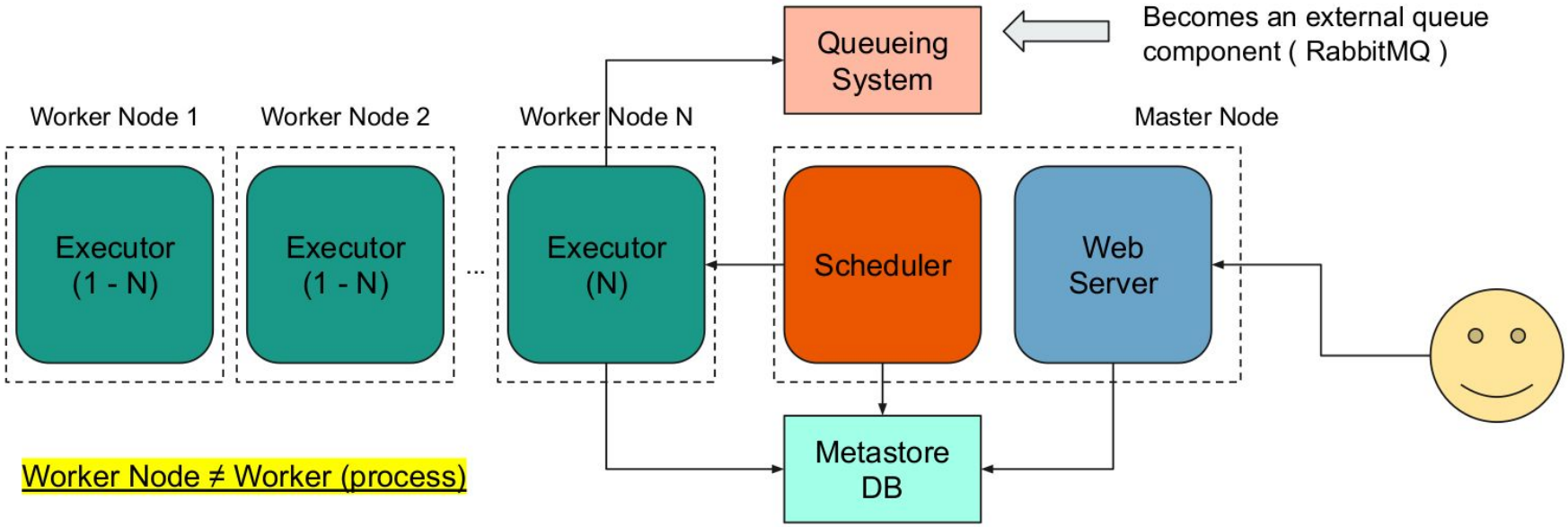
O que é!? Já aprendeu a voar?

Apache Airflow is a way to programmatically author, schedule, and monitor data pipelines

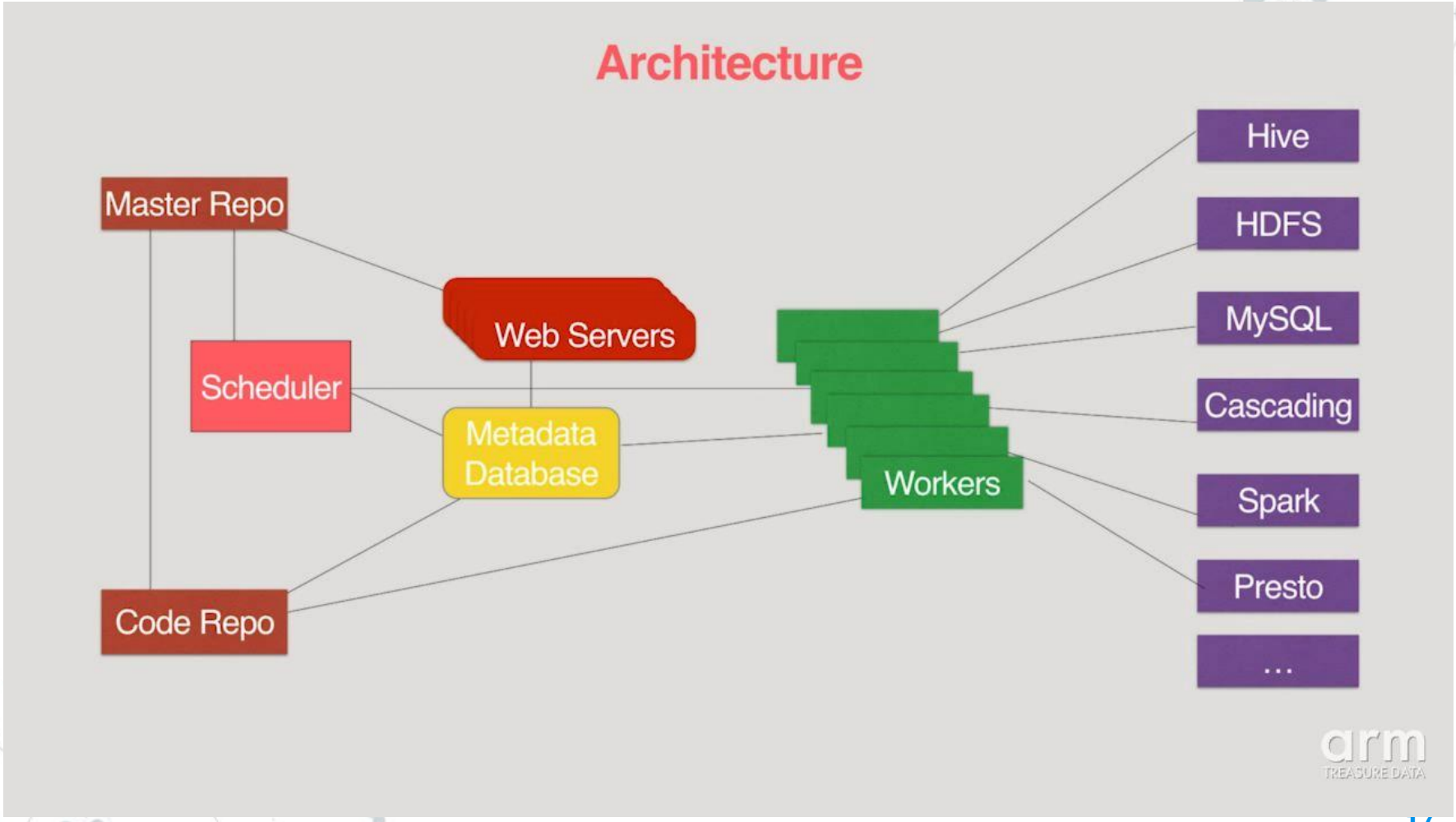
Componentes

- **WebServer**
  - UI para monitoramento dos jobs e outras coisas
- **Scheduler**
  - Server pra agendar os jobs
- **Executor**
  - É o chefe intermediário. O **Scheduler** diz pra ele o que fazer e ele delega para o **Worker**
- **Worker**
  - Executa a tarefa
- **Metadatabase**
  - Armazenamento das informações sobre as execuções das DAGs

Visão da Arquitetura



Visão da Arquitetura



Conceitos Chave

- **DAG**
  - **Grafo** que representa o **pipeline de dados**
- **Operator**
  - Descreve uma tarefa única na **DAG**
- **Task**
  - A instância do **operador**
- **Task Instance**
  - Representa uma execução específica de uma **task**
  - **= DAG + TASK + POINT IN TIME**
- **Workflow**
  - Combinação de todos os acima

O que o Airflow não é

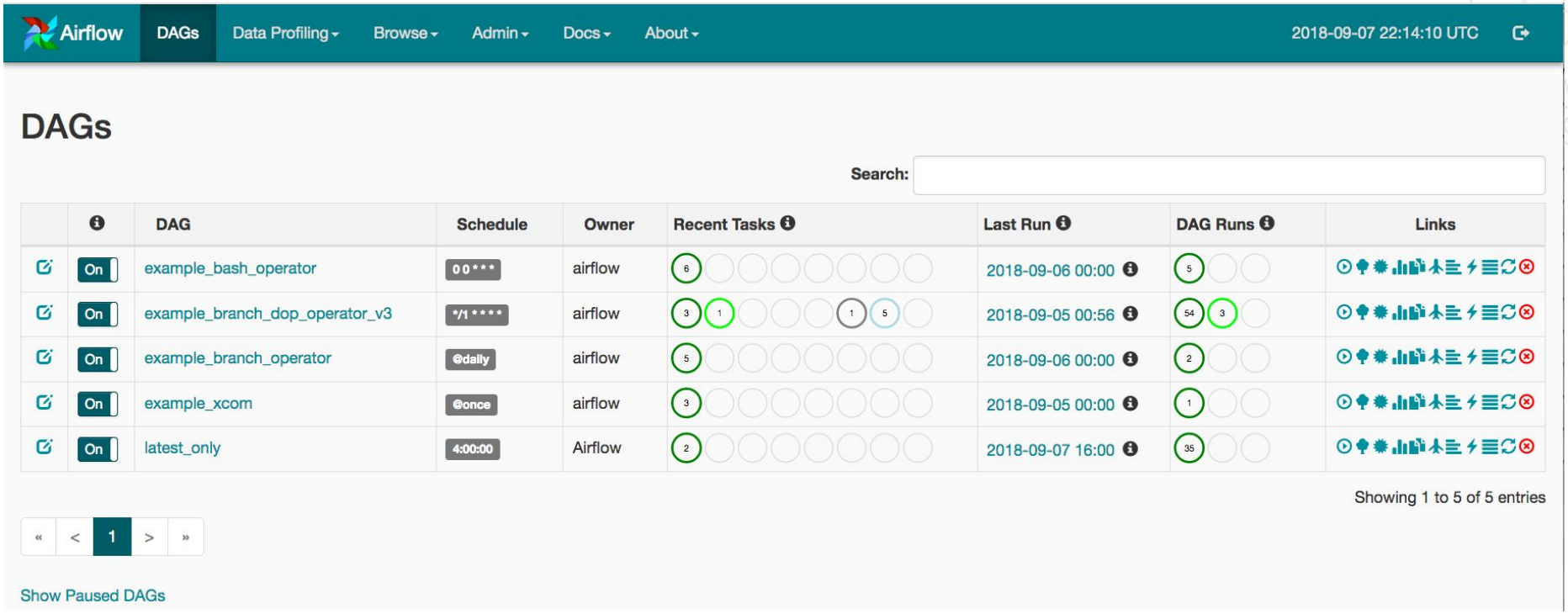
- Solução de stream de dados
  - Não está no escopo do Apache Spark ou Storm
  - Construído inicialmente para executar batch jobs agendados

3.

Airflow UI, seu lindo!

DAGs View

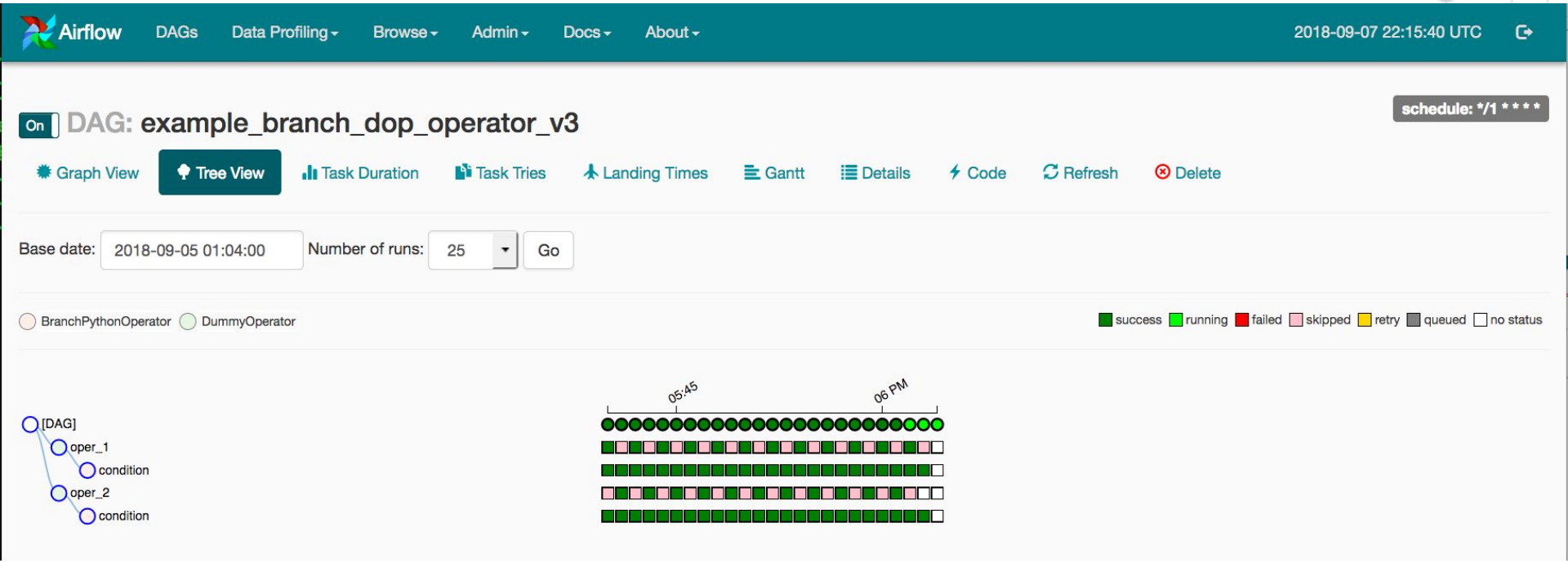
Lista dos DAGs no seu ambiente e um conjunto de atalhos para páginas úteis. Você pode ver exatamente quantas tarefas foram bem-sucedidas, falharam ou estão em execução no momento.



fonte: https://airflow.apache.org/ui.html

Tree View

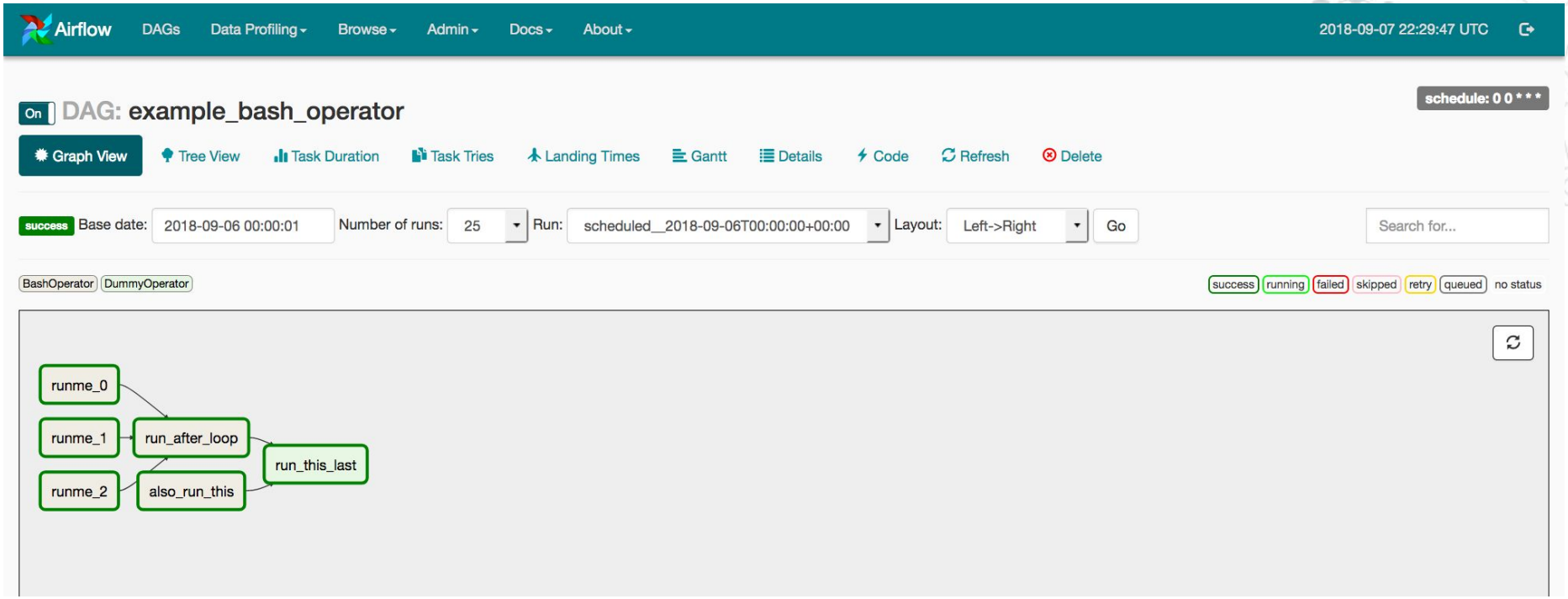
Uma representação em árvore do DAG que se estende através do tempo. Se um pipeline estiver atrasado, você poderá ver rapidamente onde estão as diferentes etapas e identificar as que estão bloqueando.



fonte: https://airflow.apache.org/ui.html

Graph View

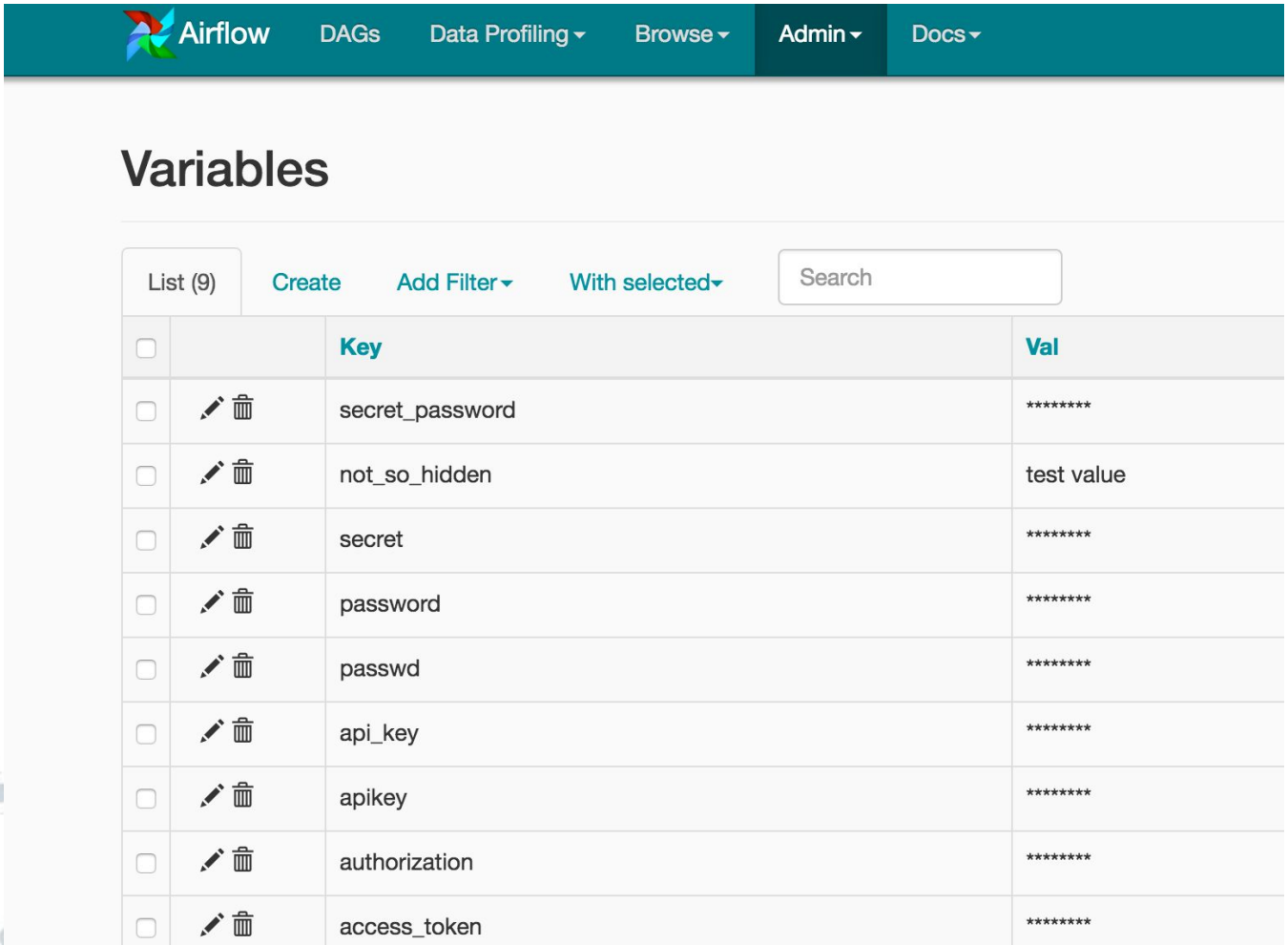
A visualização do gráfico é talvez a mais abrangente. Visualize as dependências do seu DAG e seu status atual para uma execução específica.



fonte: https://airflow.apache.org/ui.html

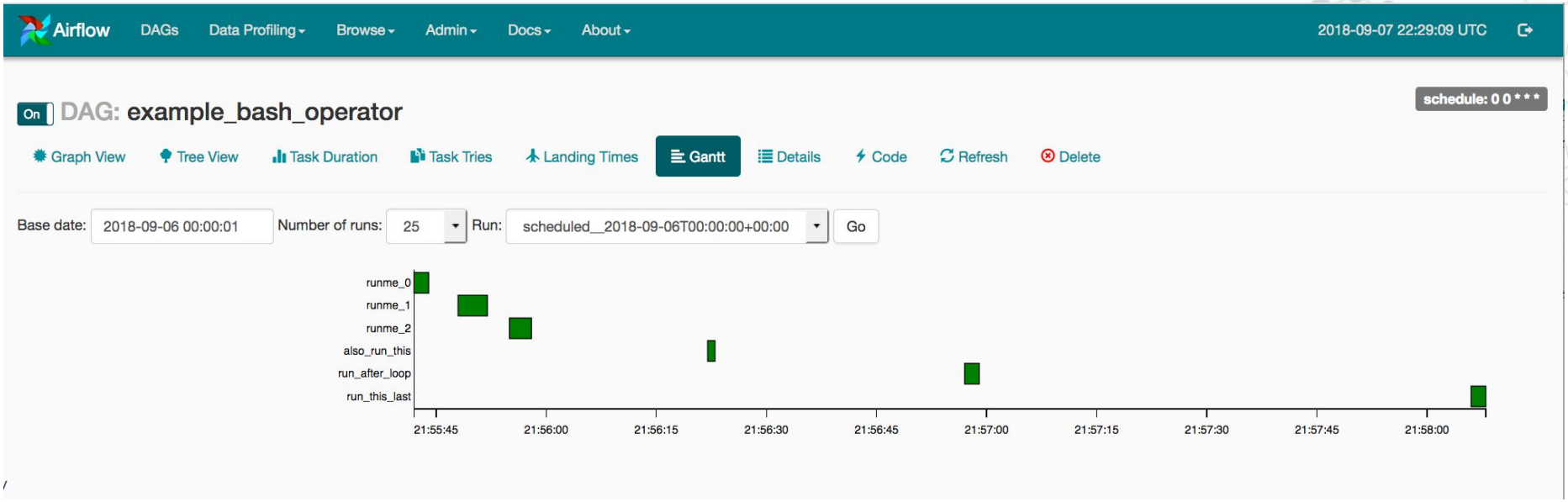
Variable View

A visualização de variável permite listar, criar, editar ou excluir o par de valores-chave de uma variável usada durante os trabalhos. O valor de uma variável ficará oculto se a chave contiver alguma palavra em ('senha', 'segredo', 'senha', 'autorização', 'api_key', 'apikey', 'access_token') por padrão, mas puder ser configurada para mostrar em texto não criptografado.



Gantt Chart

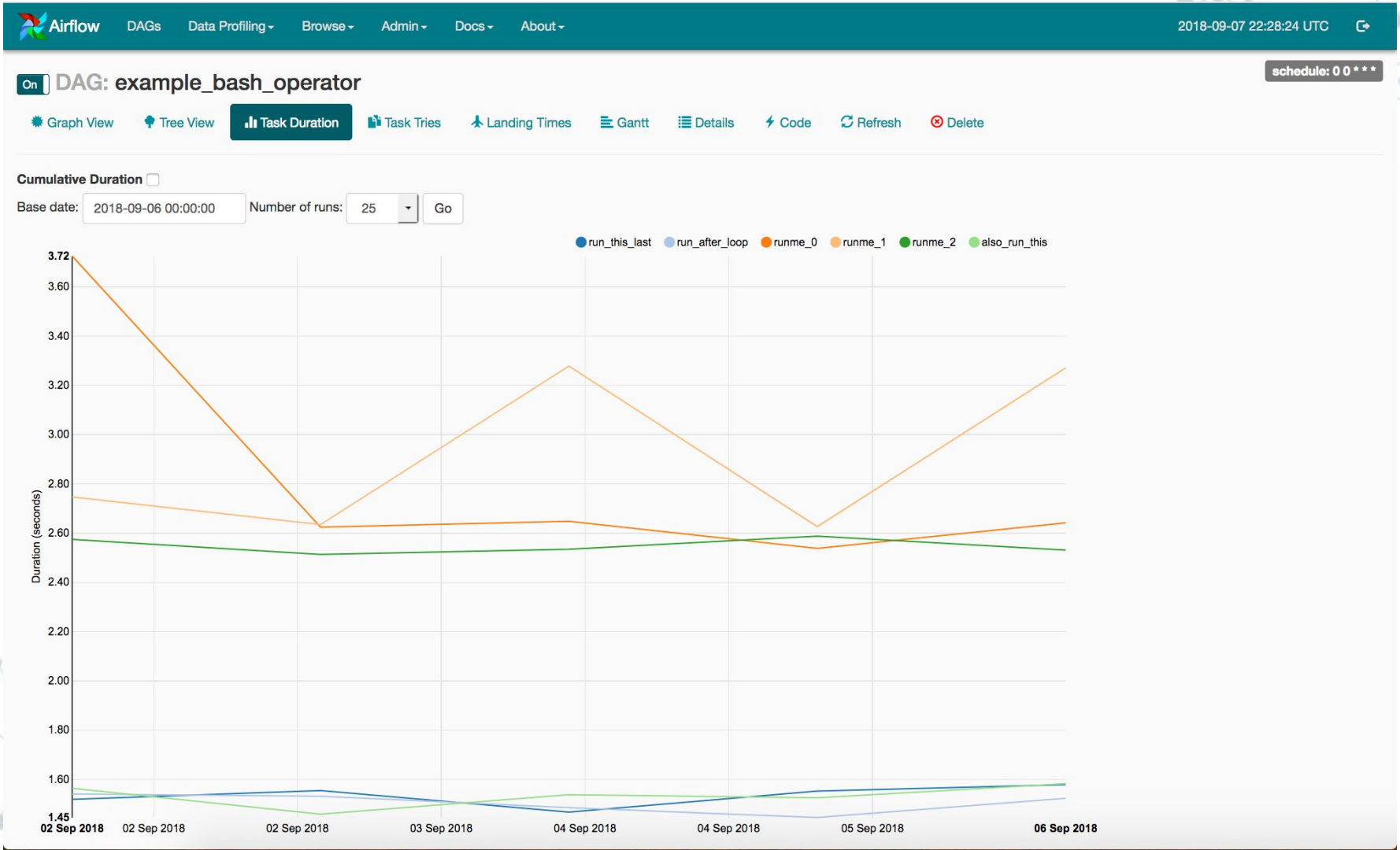
O gráfico de Gantt permite analisar a duração e a sobreposição de tarefas. Você pode identificar rapidamente gargalos e onde a maior parte do tempo é gasta para execuções específicas do DAG.



fonte: https://airflow.apache.org/ui.html

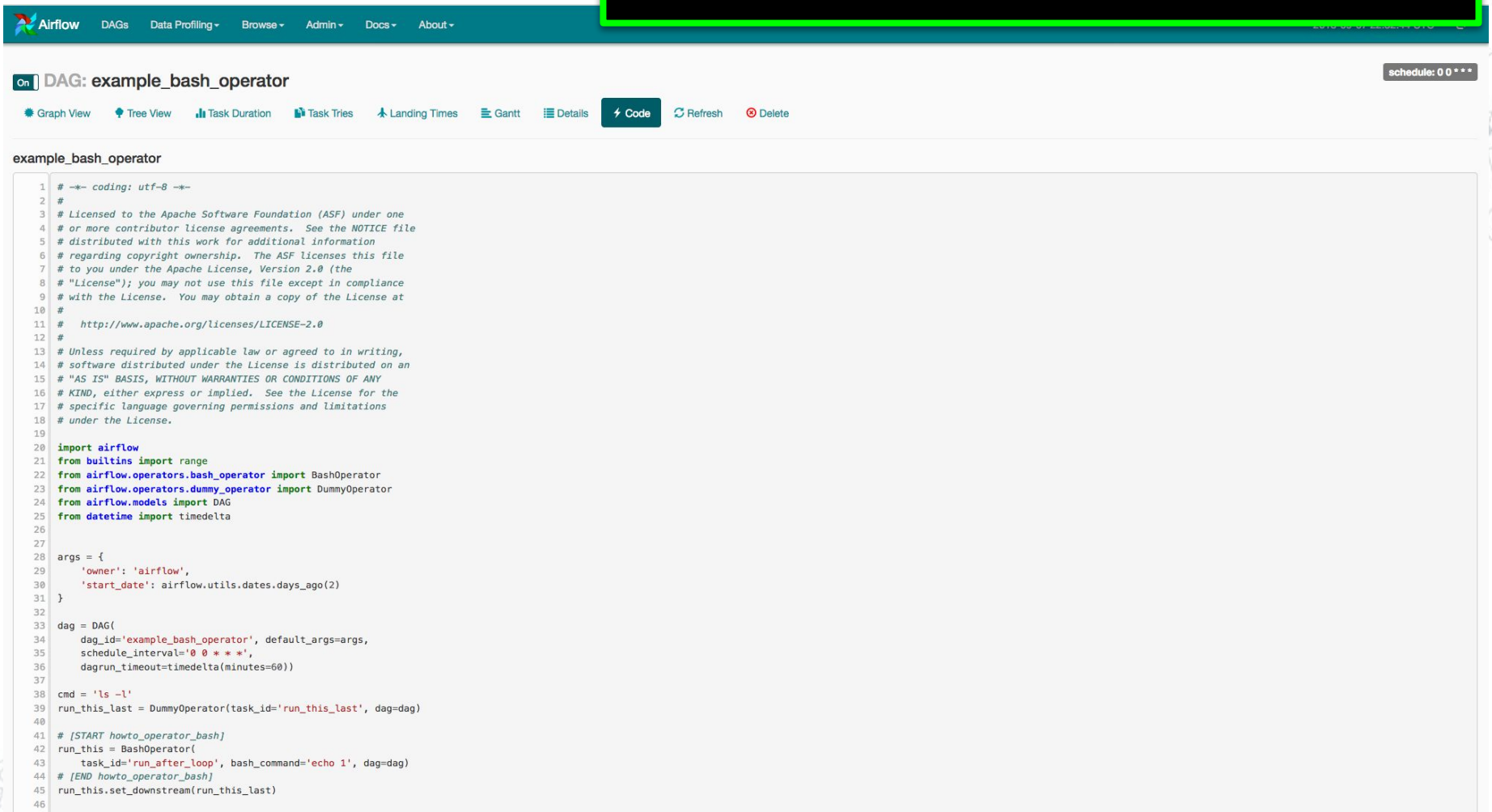
Task Duration

A duração de suas diferentes tarefas nas N corridas anteriores. Essa visualização permite encontrar discrepâncias e entender rapidamente onde o tempo é gasto no seu DAG em várias execuções.



Code View

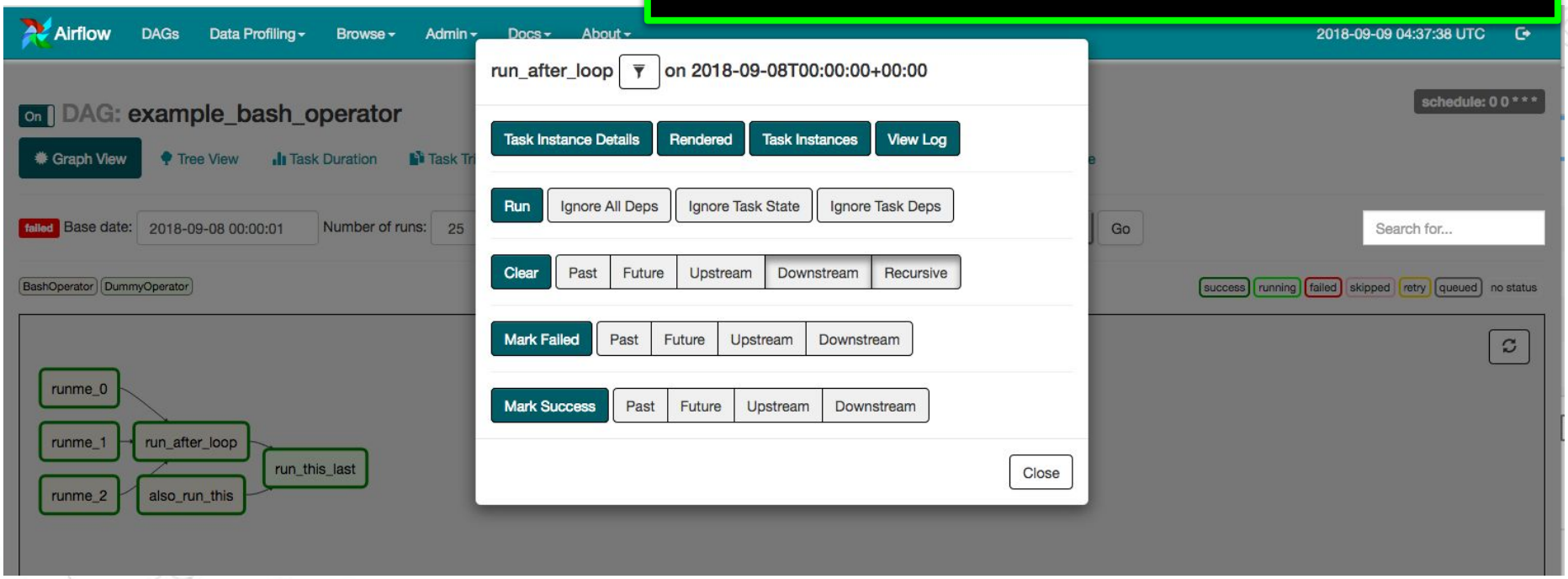
Transparência é tudo. Enquanto o código do seu pipeline está no controle de origem, esta é uma maneira rápida de acessar o código que gera o DAG e fornece ainda mais contexto.



fonte: https://airflow.apache.org/ui.html

Task Instance Context Menu

Nas páginas vistas acima (exibição em árvore, exibição de gráfico, gantt,...), sempre é possível clicar em uma instância de tarefa e acessar esse menu de contexto avançado que pode levá-lo a metadados mais detalhados e executar algumas ações.



fonte: https://airflow.apache.org/ui.html

4.

DAG: que danado é isso!?

Definição de DAG

- Conceitualmente uma DAG (Grafo Acíclico Direcionado) é um **grafo direcionado** finito que não possui ciclos (loops).

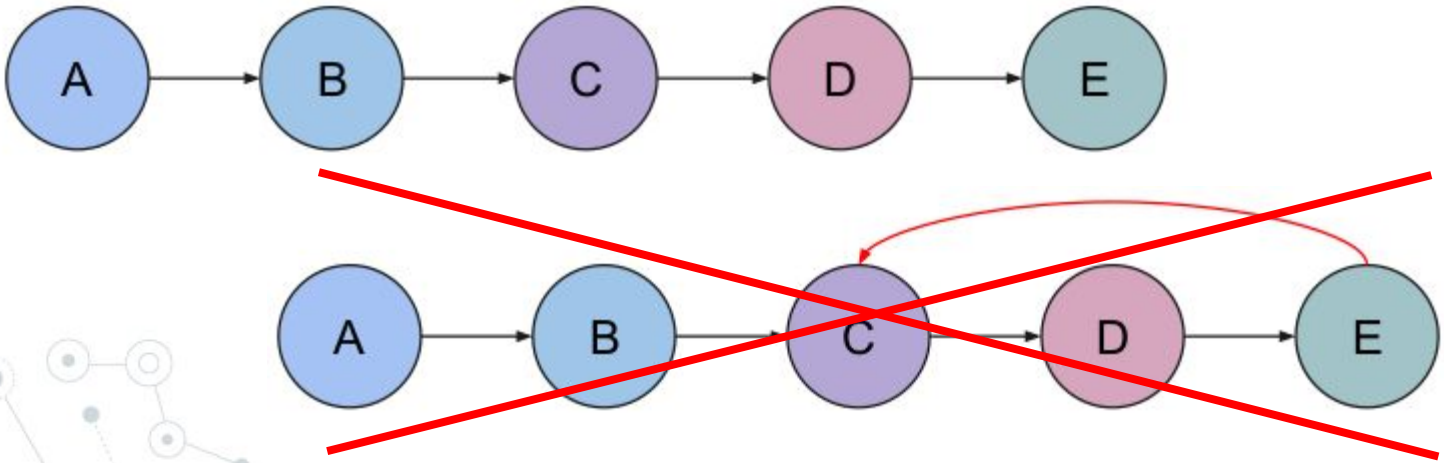
- No Airflow, uma DAG representa um conjunto de **tarefas** a serem executadas, organizadas de maneira a representar suas dependências e relacionamentos.

  - Cada Nó é uma tarefa

  - Cada Aresta é uma dependência

Um exemplo fictício simples

- Nó A faz download de arquivo de dados
- Nó B envia os dados para processamento
- Nó C monitora e aguarda conclusão do processamento
- Nó D atualiza indicador no BD
- Nó E envia notificação de conclusão via Telegram



O que são Operadores?

- Um operador descreve uma única Tarefa em um Workflow
- Geralmente os operadores são atômicos
  - Não compartilham dados com outro operador
- São reexecutados em caso de falhas
- A DAG garante executar os operadores na ordem correta

Operadores nativos do Airflow

- BashOperator
  - Executa um comando Bash
- PythonOperator
  - Chama uma função python qualquer
- EmailOperator
  - Envia um email
- MySqlOperator, SqliteOperator, PostgreOperator…
  - Executa um comando SQL

Tipos de Operadores

Todos Operadores herdam de *BaseOperator*.

Existem 3 tipos de operadores:

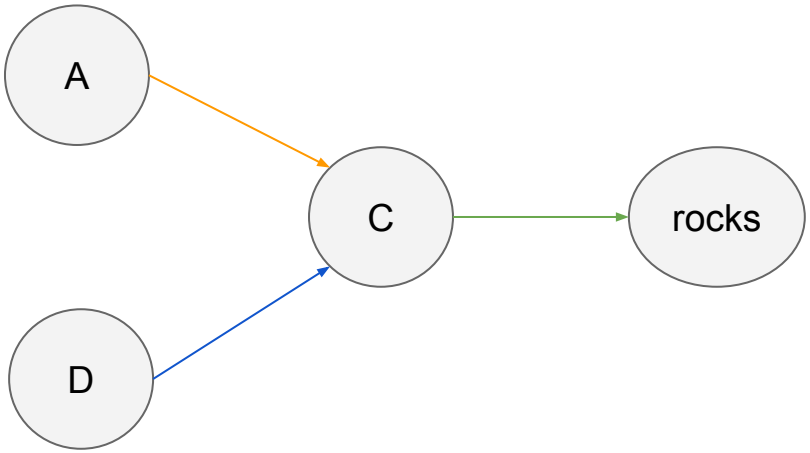
- Operadores de **Ação**
  - (BashOperator, PythonOperator, EmailOperator…)
- Operadores de **Transferência**
  - (MongoToS3Operator, MySqlToHiveTransfer, SFTPOperator…)
- Operadores **Sensores**
  - (SqlSensor, HttpSensor, HdfsSensor…)

Operadores pra todo gosto...

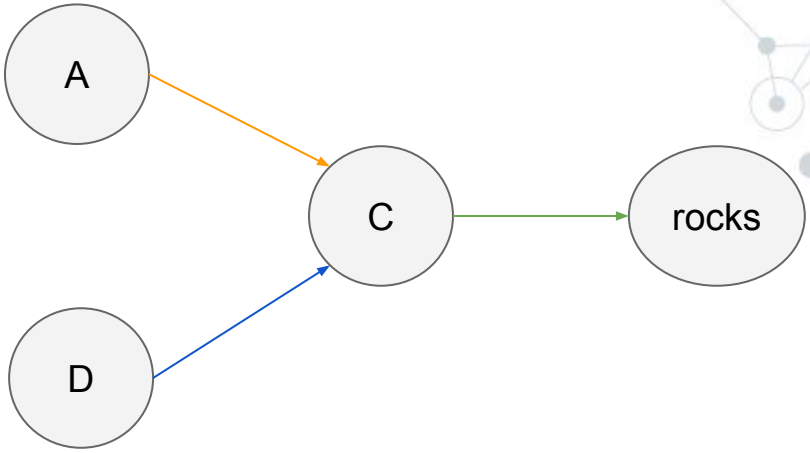
- airflow.operators.bash_operator
- airflow.operators.branch_operator
- airflow.operators.check_operator
- airflow.operators.dagrun_operator
- airflow.operators.docker_operator
- airflow.operators.druid_check_operator
- airflow.operators.dummy_operator
- airflow.operators.email_operator
- airflow.operators.generic_transfer
- airflow.operators.hive_operator
- airflow.operators.hive_stats_operator
- airflow.operators.hive_to_druid
- airflow.operators.hive_to_mysql
- airflow.operators.hive_to_samba_operator
- airflow.operators.http_operator
- airflow.operators.jdbc_operator
- airflow.operators.latest_only_operator
- airflow.operators.mssql_operator
- airflow.operators.mssql_to_hive
- airflow.operators.mysql_operator
- airflow.operators.mysql_to_hive
- airflow.operators.oracle_operator
- airflow.operators.pig_operator
- airflow.operators.postgres_operator
- airflow.operators.presto_check_operator
- airflow.operators.presto_to_mysql
- airflow.operators.python_operator
- airflow.operators.redshift_to_s3_operator
- airflow.operators.s3_file_transform_operator
- airflow.operators.s3_to_hive_operator
- airflow.operators.s3_to_redshift_operator
- airflow.operators.sensors
- airflow.operators.slack_operator
- airflow.operators.sqlite_operator
- airflow.operators.subdag_operator
- airflow.contrib.operators.bigquery_to_bigquery
- airflow.contrib.operators.bigquery_to_gcs
- airflow.contrib.operators.cassandra_to_gcs
- airflow.contrib.operators.databricks_operator
- airflow.contrib.operators.dataflow_operator
- airflow.contrib.operators.dataproc_operator
- airflow.contrib.operators.datastore_export_operator
- airflow.contrib.operators.datastore_import_operator
- airflow.contrib.operators.dingding_operator
- airflow.contrib.operators.discord_webhook_operator
- airflow.contrib.operators.druid_operator
- airflow.contrib.operators.ecs_operator
- airflow.contrib.operators.emr_add_steps_operator
- airflow.contrib.operators.emr_create_job_flow_operator
- airflow.contrib.operators.emr_terminate_job_flow_operator
- airflow.contrib.operators.file_to_gcs
- airflow.contrib.operators.file_to_wasb
- airflow.contrib.operators.gcp_bigtable_operator
- airflow.contrib.operators.gcp_cloud_build_operator
- airflow.contrib.operators.gcp_compute_operator
- airflow.contrib.operators.gcp_container_operator
- airflow.contrib.operators.gcp_dlp_operator
- airflow.contrib.operators.gcp_function_operator
- airflow.contrib.operators.gcp_natural_language_operator

Compondo Tarefas

- Nesta DAG, os nós são A, D, C e rocks
- As arestas são as setas coloridas
- Cada seta representa uma dependência
  - **C** depende de **A** e **D**
  - **rocks** depende de **C**



Exemplo:



As dependências podem ser expressas de 4 formas:

- `A.set_downstream(C); D.set_downstream(C); D.set_downstream(rocks)`
- `rocks.set_upstream(C); C.set_upstream(A); C.set_upstream(D)`
- `rocks << C; C << A; C << D`
- `A >> C; D >> C >> rocks`

Parâmetros da DAG

- `start_date`
  - A primeira data para a qual você deseja que os dados sejam produzidos pela DAG (pode ser definido no passado)
- `end_date`
  - A data em que sua DAG deve parar de ser executado (none por padrão)
- `retries`
  - O número máximo de tentativas antes da Task falhar
- `retry_delay`
  - O tempo de espera até a tentativa seguinte
- `schedule_interval`
  - O intervalo em que o Scheduler acionará sua DAG

schedule_interval

- Indica em qual intervalo o Scheduler deve rodar a sua DAG
- Aceita expressões CRON, `datetime.timedelta` ou presets.

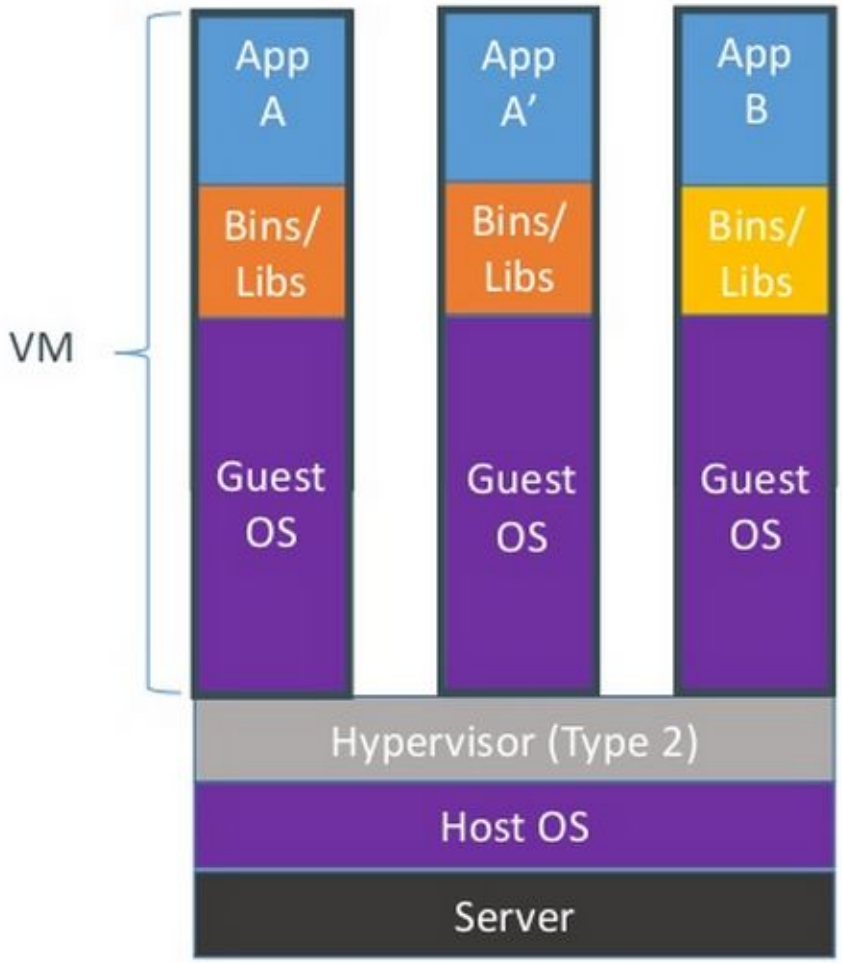
Predefinidos	Resultado	Cron
None	Não agenda. Acionado manualmente	
@once	Agenda uma e somente uma vez	
@hourly	Roda uma vez por hora no início da hora	0 * * * *
@daily	Roda uma vez por dia à meia-noite	0 0 * * *
@weekly	Roda uma vez por semana à meia-noite no domingo cedo	0 0 * * 0
@monthly	Roda uma vez por mês à meia-noite do primeiro dia do mês	0 0 1 * *
@yearly	Roda uma vez por ano à meia-noite de 1º de Janeiro	0 0 1 1 *

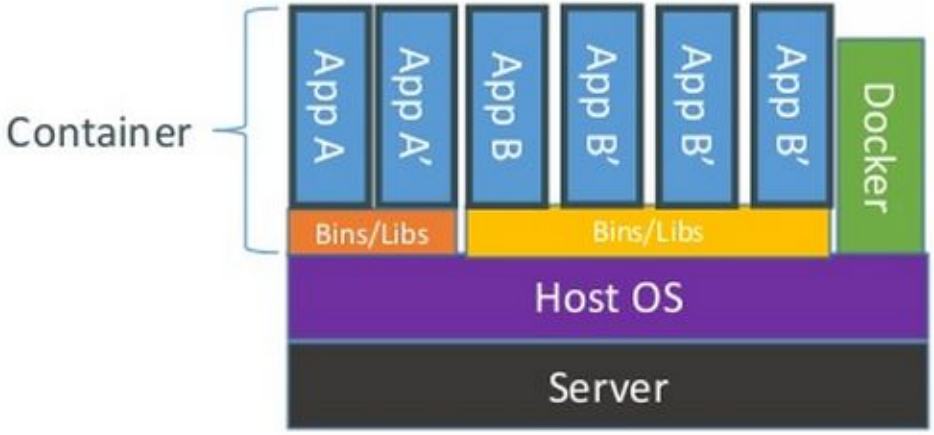
5.

Baleia intro

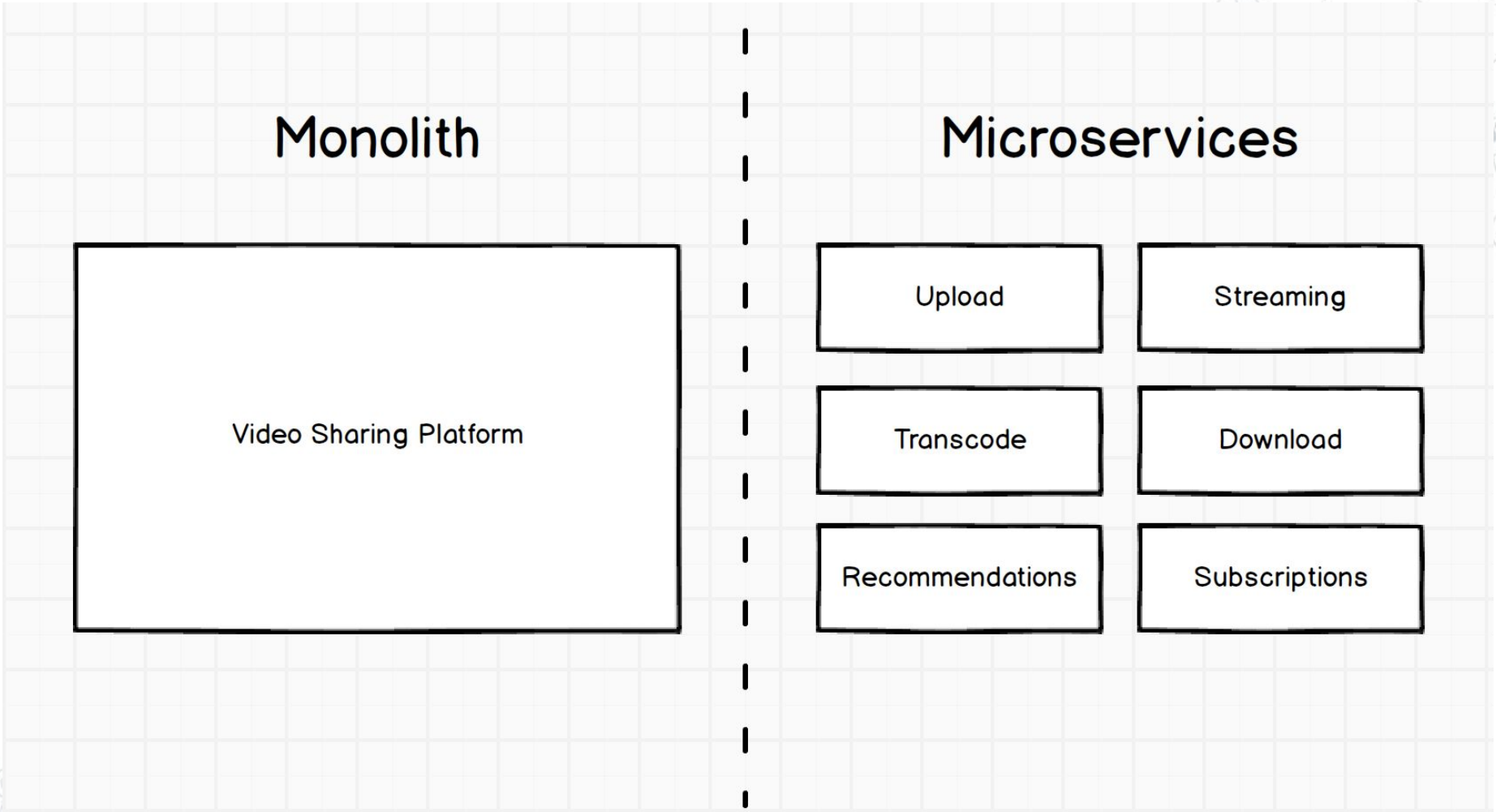
(Docker)

Containers vs VMs





Monolítico vs Microserviços



Venha ser feliz com Docker!

- Isolamento
- Execução leve
- Simplicidade
- Workflow DevOps
- Comunidade

6.
Botar pra rodar

Pool

Levanta a mão quem não tem o docker instalado na máquina

```
if 🙌 and SO == “Linux”: #eeeeee
```

https://docs.docker.com/install/

```
elif 🙌 and SO == “Windows”: #uuuu
```

https://docs.docker.com/docker-for-windows/install/

```
else:
   print(“Suave na nave”)
```

Github

No terminal:

```
$ git clone
https://github.com/economiagovbr/oficina-airflow.git
```

(também conhecido como download com sincronização)

economiagovbr / oficina-airflow

Nitai Bezerra da Silva 'Escondendo' a solução, apenas alterando o formato para o Airflow não… ···		Latest commit `545209f` 1 hour ago
dags	'Escondendo' a solução, apenas alterando o formato para o Airflow não…	1 hour ago
docker-airflow	add docker	3 days ago
.gitignore	cria arquivos desafio e solucao	23 hours ago
README.md	Initial commit	3 days ago

Branch: **master** ▾ **oficina-airflow** / **dags** /

Create new file | **Upload files** | **Find file** | **History**

Nitai Bezerra da Silva 'Escondendo' a solução, apenas alterando o formato para o Airflow não… ···		Latest commit `545209f` 2 hours ago
..		
desafio	atualiza desafio_baby_steps_dag.py	23 hours ago
export	Renomenando arquivo para não confundir	19 hours ago
solucao	'Escondendo' a solução, apenas alterando o formato para o Airflow não…	2 hours ago
sql_query	cria arquivos desafio e solucao	23 hours ago
tuto.py	prep docker e dados indigenas	3 days ago

Github

No terminal:

```
$ cd oficina-airflow/docker-airflow
$ docker-compose up
```

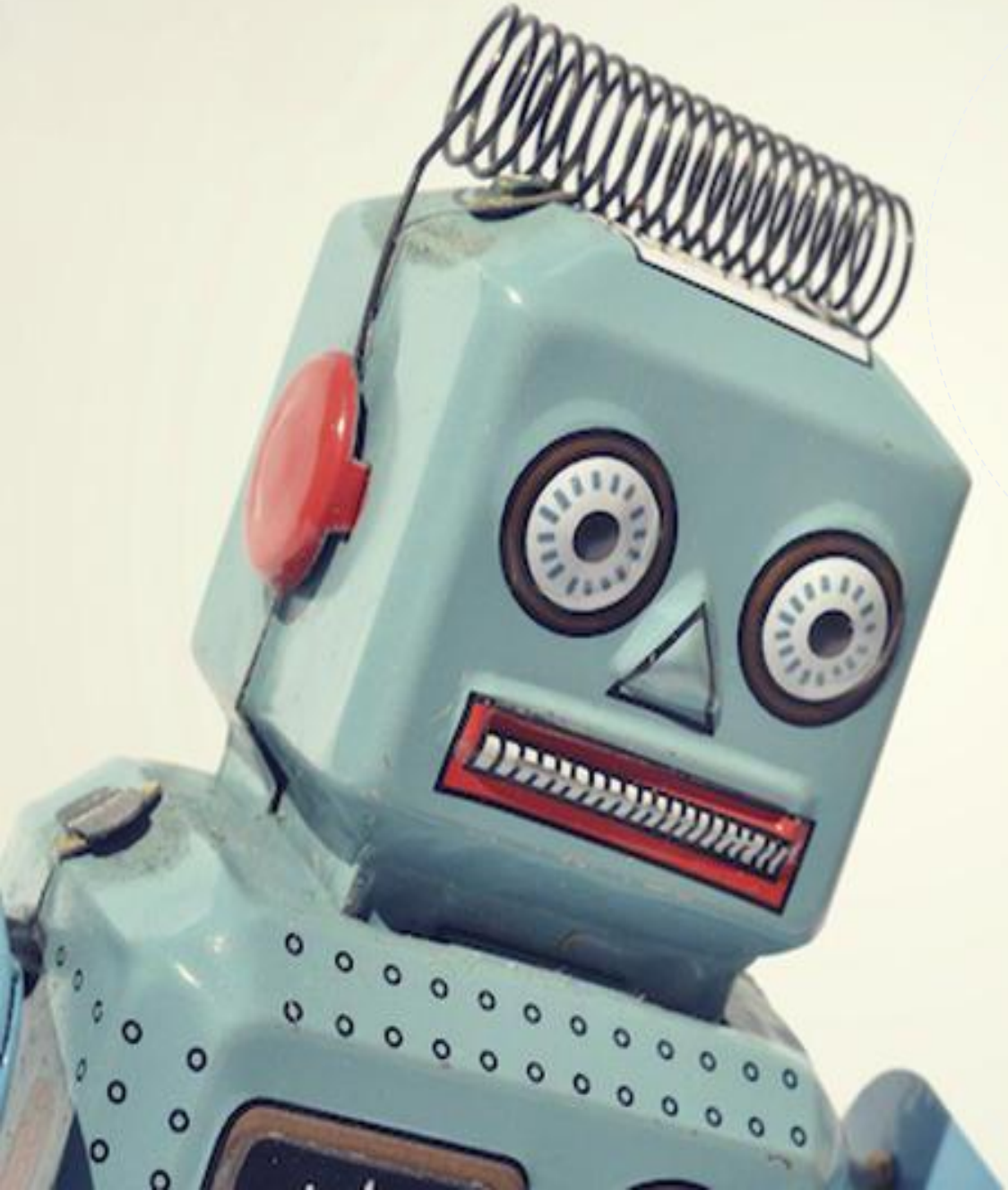
(também conhecido como: magia)

Se funcionou sua tela está assim

Agora foi

Acesse pelo navegador:

http://localhost:8080/

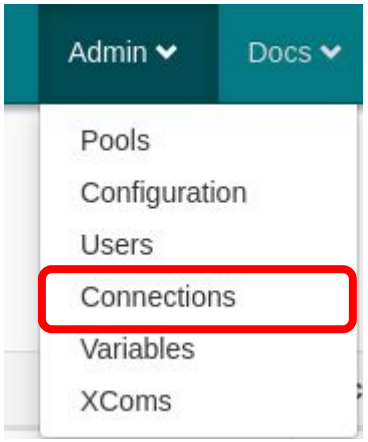


7.

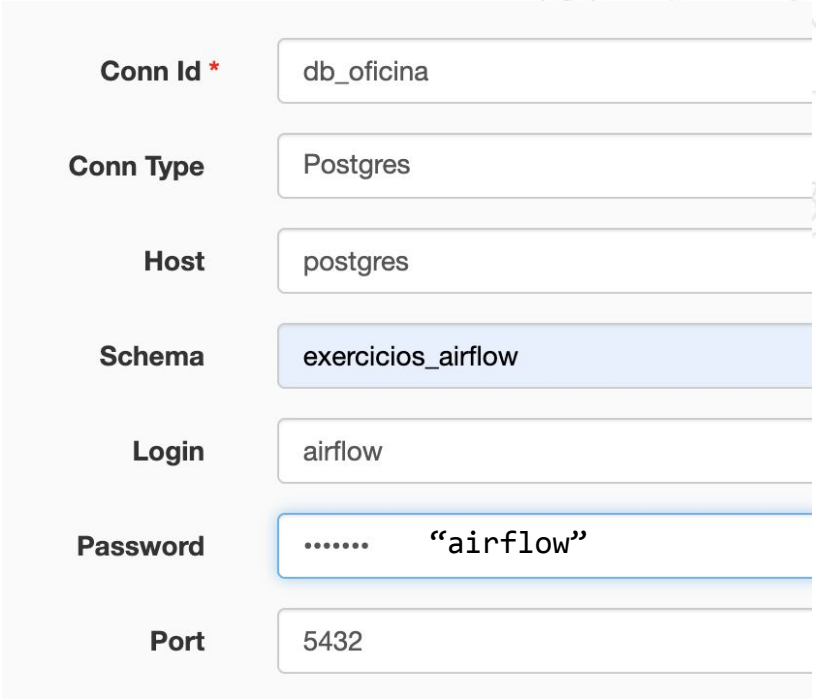
SELECT test FROM oficina

SELECTzinho básico pra conferir se está tudo ok

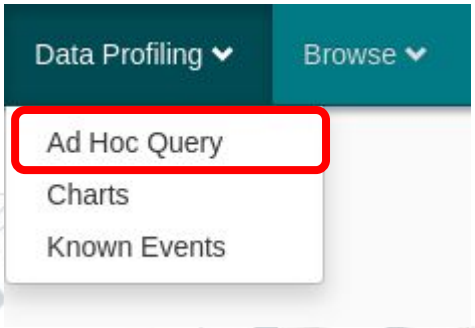
1. Criar conexão



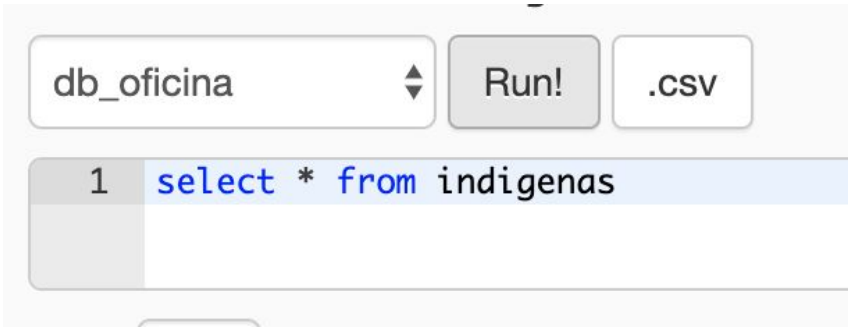
>>



2. Executar a Query



>>



Origem dos dados

- **Terras Indígenas - estados - 2017.csv**
  - http://dados.gov.br/dataset/sistema-nacional-de-informacoes-florestais-snif/resource/6086a36e-d5dc-4fea-b2e0-a9d1d3229867

8.

[break] Ufa, DESCANSO (20 min.)

8.

[break] Ufa, DESCANSO (20 min.)

9.

Hello World: Seu primeiro robô (ou não)

Orientações

1. Edite o arquivo **`desafio/desafio_hello_world_dag.py`** seguindo as instruções nos comentários

2. A solução com o script python está em **`solucao/solucao_hello_world_dag.py.sol`**, mas guarde esse segredo ;P

Estrutura de uma DAG

```
# Libraries
# Tudo ok
...
# DAG
# Definir parâmetros
...
# Tasks
# Set nome e comando bash
...
# Orchestration
# Ordem de execução
...
```

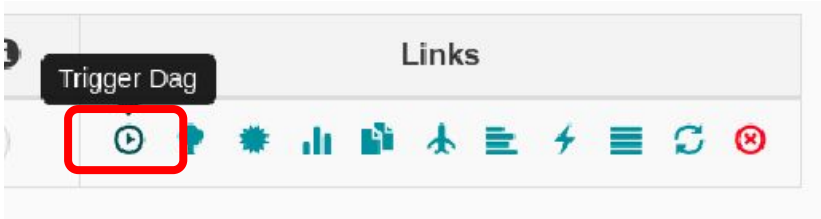
Prática:

Implementar uma DAG com 2 operadores Bash que imprimem "Hello" e "World!"

Para dar o Play

1. No painel do Airflow, ligue a DAG:

2. No painel, nos botões “Links”, acione o “Trigger Dag”:



10.

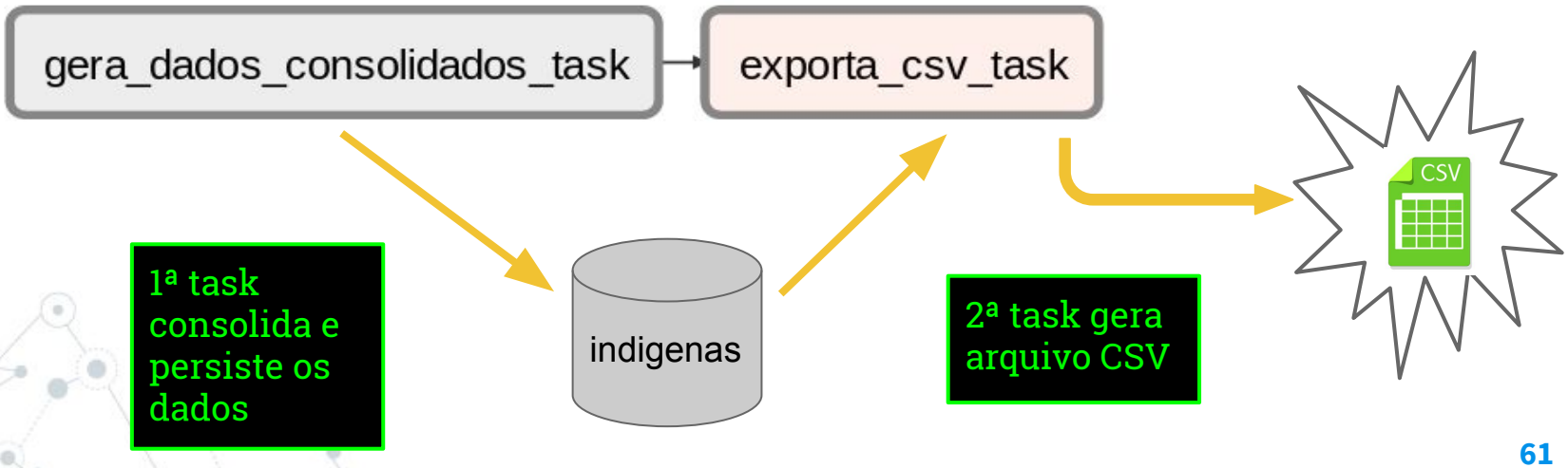
Baby Steps: Implementação de uma DAG pra chamar de sua

Desafio firmeza

1. Edite o arquivo **`desafio/desafio_baby_steps_dag.py`** seguindo as instruções nos comentários

2. Edite o arquivo **`sql_query/desafio_query_uf_superficie_total.sql`** seguindo as instruções nos comentários

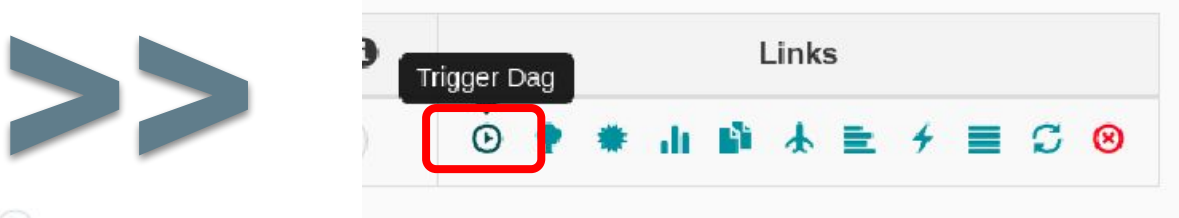
3. A solução com o script python está em **`solucao/solucao_baby_steps_dag.py.sol`**, mas guarde esse segredo ;P



Para dar o Play

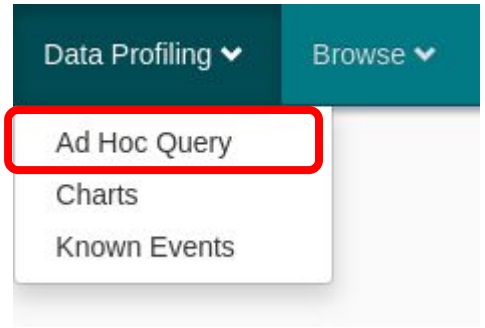
1. No painel do Airflow, ligue a DAG:

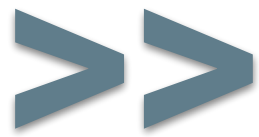
2. No painel, nos botões “Links”, acione o “Trigger Dag”:

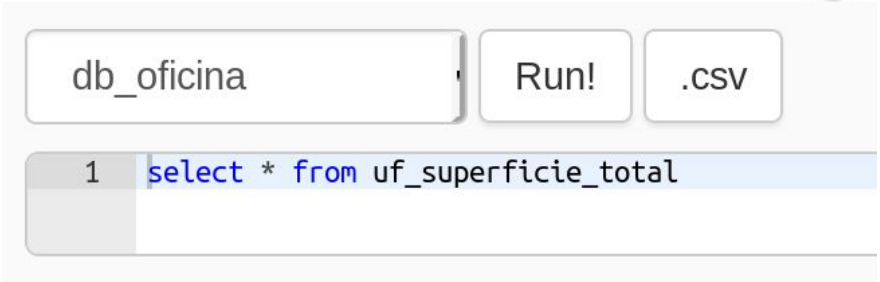


Se pá deu certo

1. Verificar se os dados foram gerados. Execute a query:





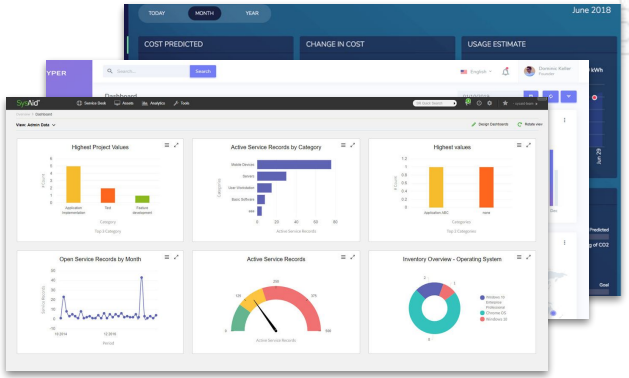


2. Verificar se o arquivo CSV foi gerado. Acesse a pasta "**`export/`**" e procure pelo arquivo **`.csv`**

11.
Cataventos do ME

Somos a Coordenação-Geral de Gestão da Informação/SEGES

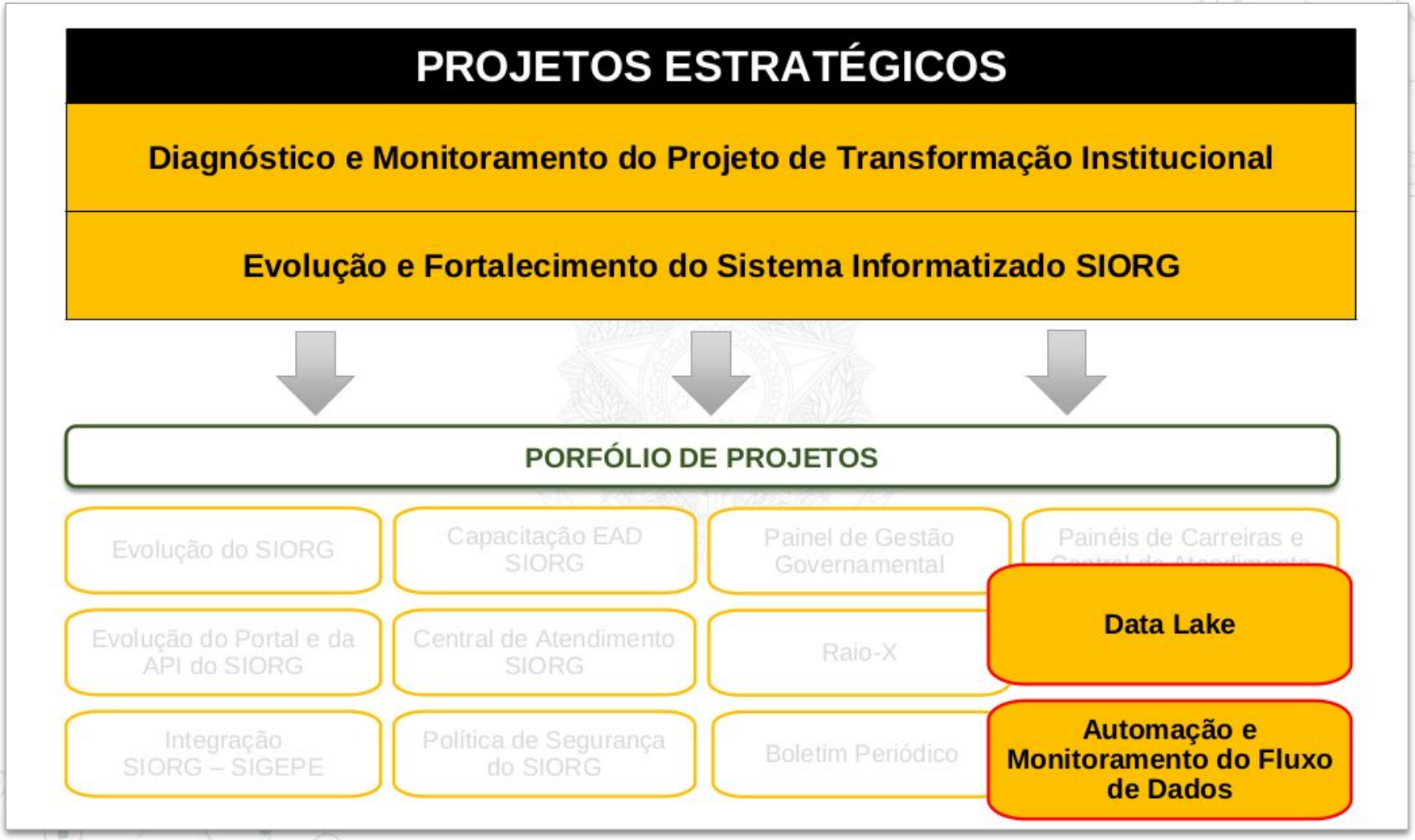
- Fazemos apoio à gestão estratégica
- Painéis informacionais
- Integrando dados estruturantes
- Mais de 2 anos produzindo



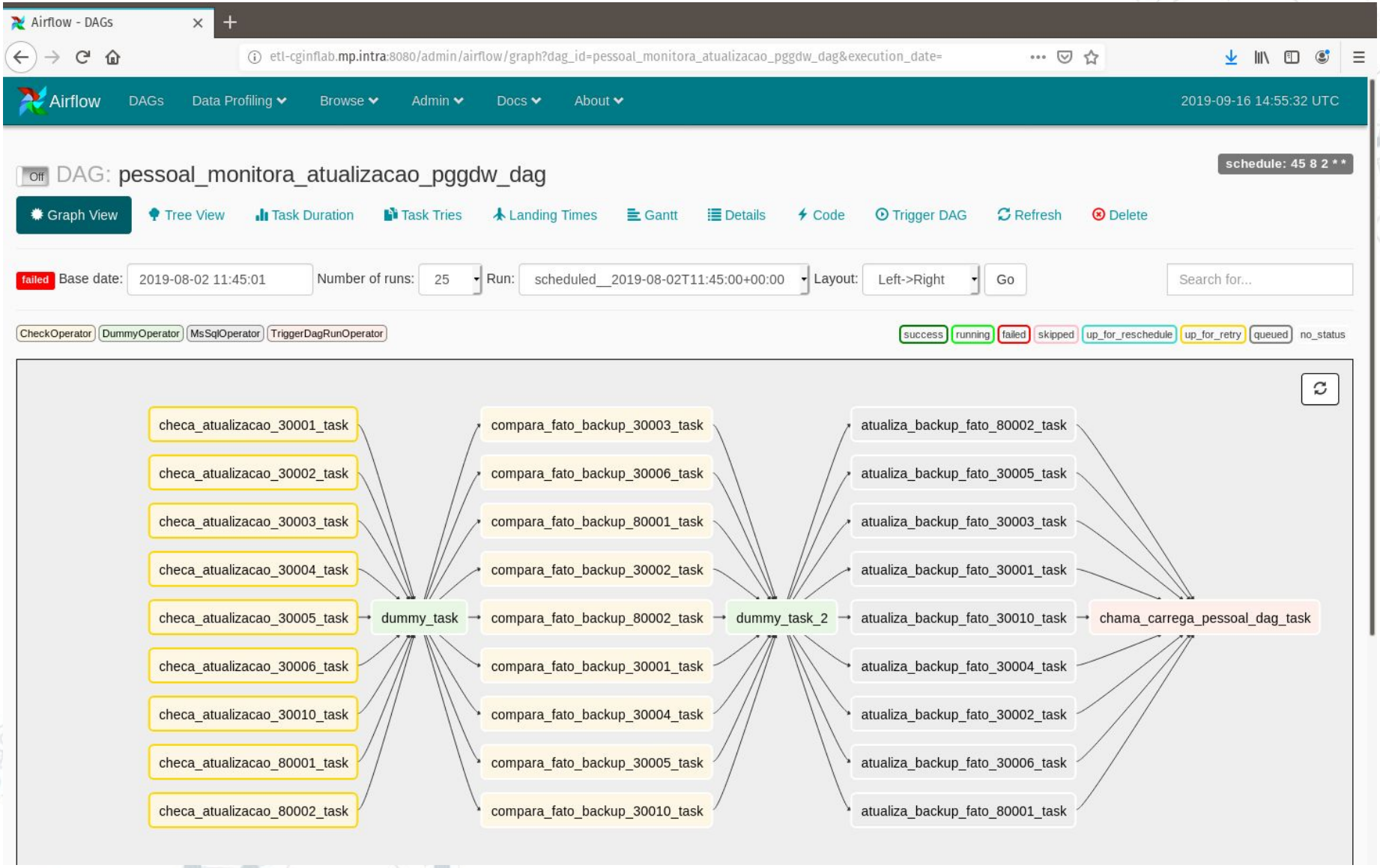
Acumulamos uma dívida técnica grande no backend.

😭 😑 😖

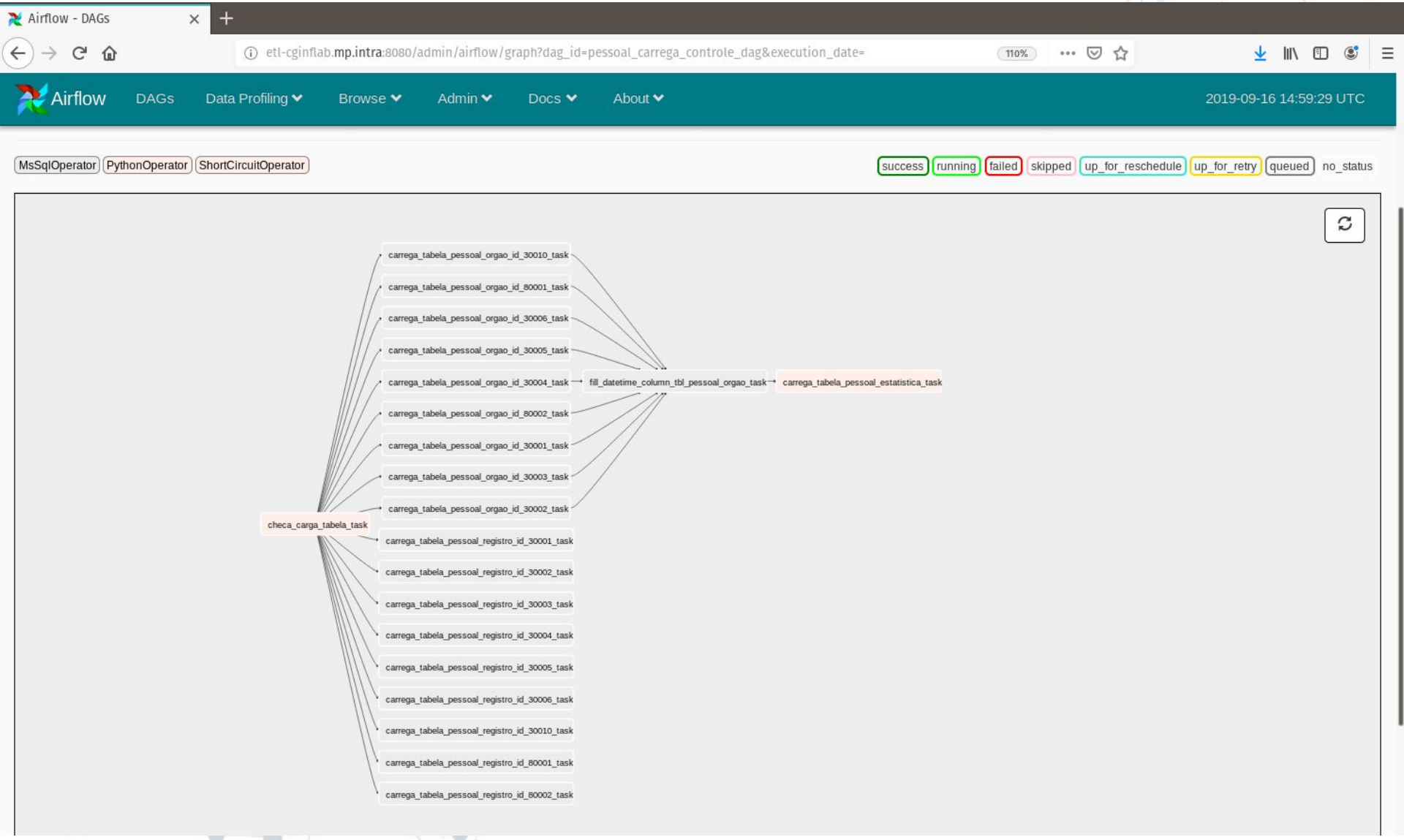
Convertendo o Chefe



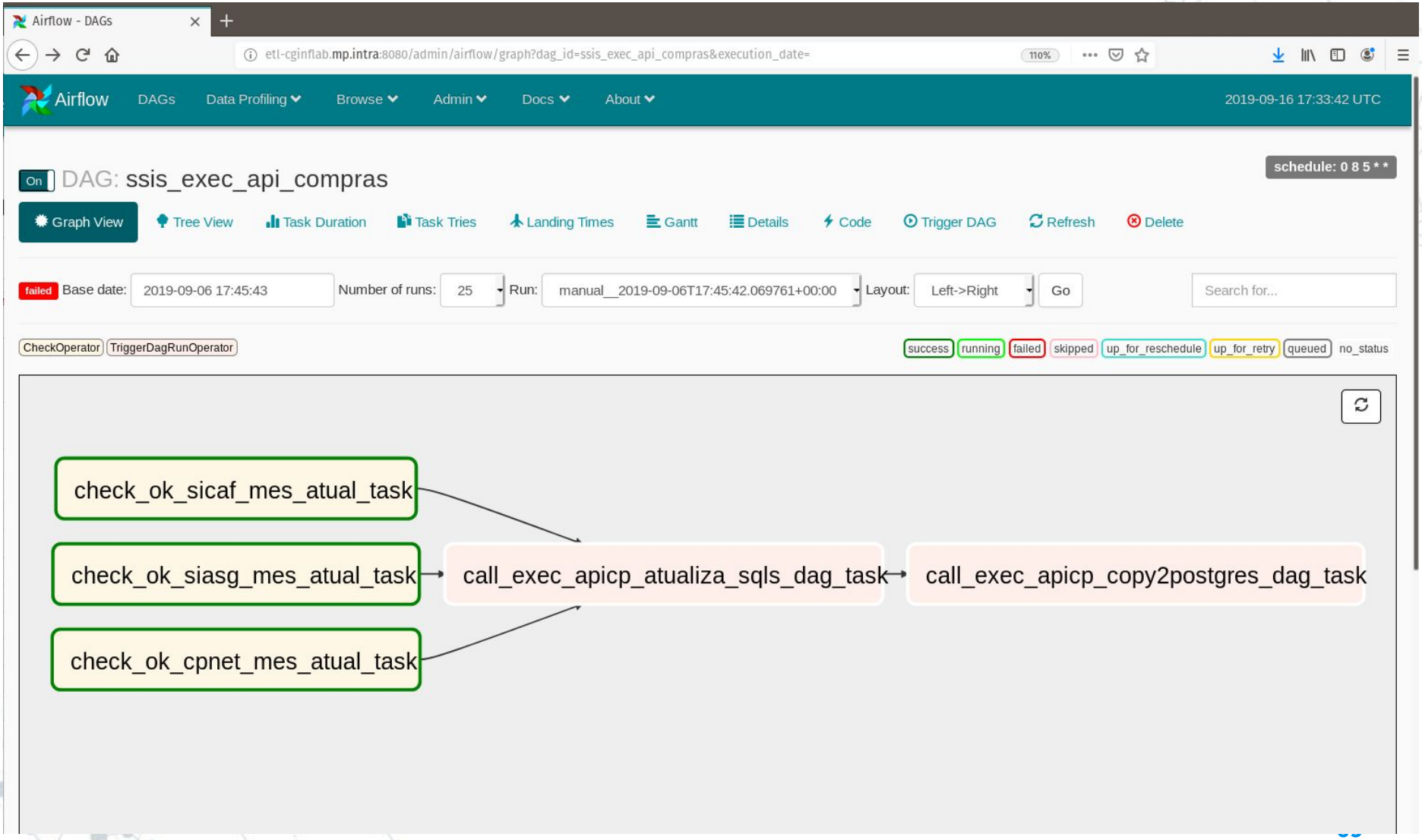
Monitoramento e visualização do ETL Legado



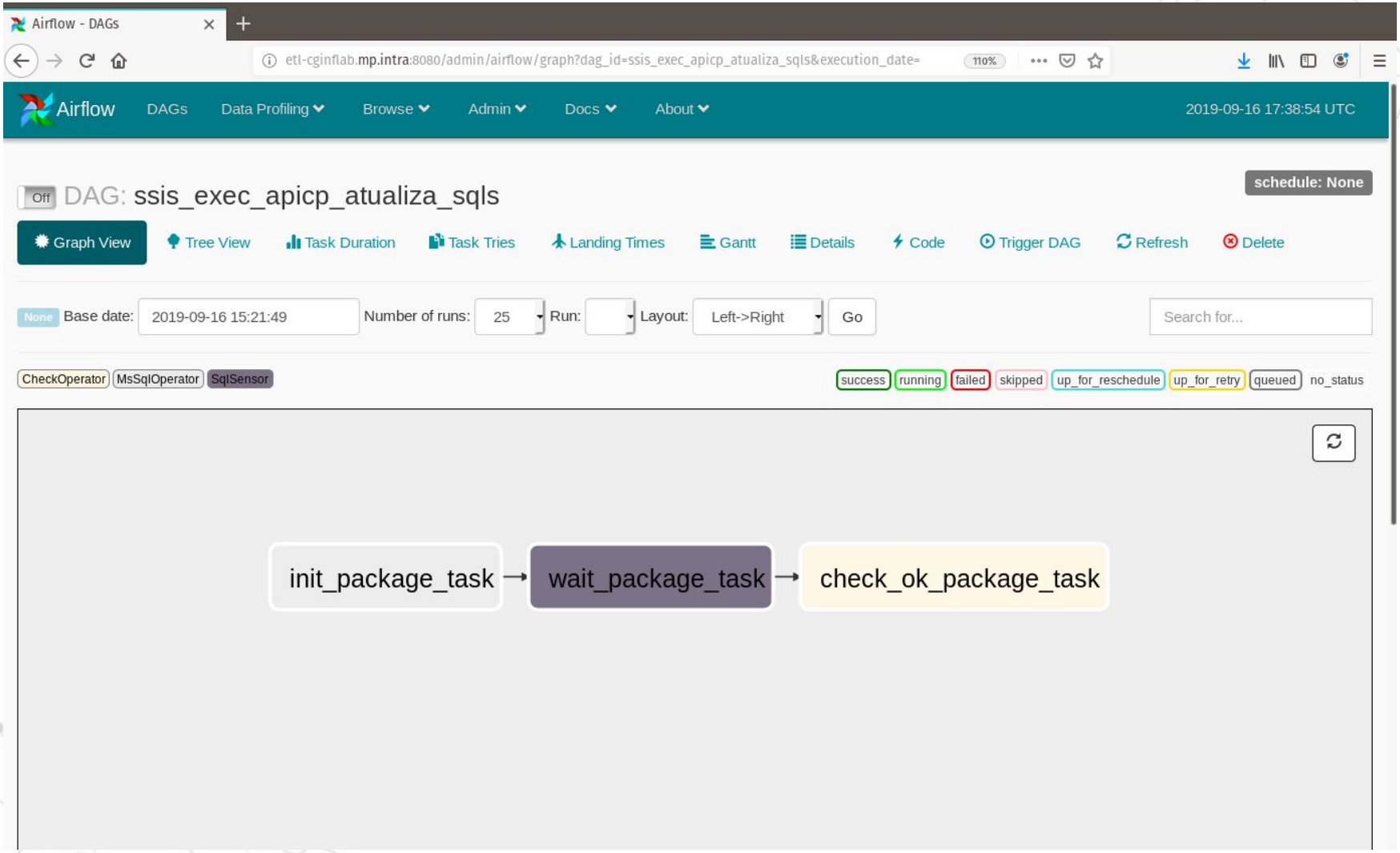
Monitoramento e visualização do ETL Legado



Orquestração da carga da API de Compras



DAG secundária



12.
Dicas e Truques (quem souber me conta)

Reflexões de arquitetura

- Papel do Airflow como Orquestrador e não como motor de ETL
- Uma DAG enorme **vs** Várias pequenas DAGs
- Caminhos seguros para transição
- Geração automática de DAGs
- PythonOperator **vs** Custom Operators

Treinamentos

- The Complete Hands-On Course to Master Apache Airflow - Udemy/Marc Lamberti
- Apache Airflow Tutorials - Apply Datascience

Nuvens com Airflow PAAS

- Google Cloud Composer
  https://cloud.google.com/composer/
- Astronomer.io

Manual e manuéis

- Manual Oficial Apache Airflow
- Github Apache Airflow
- Puckel/docker-airflow (Github)
- ETL best practices with Airflow documentation site - Gerard Toonstra
- Best practices with Airflow- an open source platform for workflows & schedules - Maxime Beauchemin
- Hands-On Big Data Online Courses - Marc Lamberti

ecodados.economia.gov.br

Ecodados - Fórum dos cien

https://ecodados.economia.gov.br

Ecodados

Para facilitar o lançamento do seu novo site, você está no modo de bootstrap. Todos os novos usuários receberão o nível de confiança 1 e terão os e-mails diários de resumo ativados. Isso será desativado automaticamente quando 100 usuários se registrarem.

todas as categorias | Recente | Melhores | Categorias | + Novo Tópico

Tópico	Respostas	Visualizações	Atividade
Scheduling Notebooks at Netflix medium.com **Artigos e Publicações** jupyter, airflow, notebooks	1	7	4d
Voltamos! ou: renovação do certificado digital **Equipe** downtime, lets-encrypt	3	6	6d
Base de Acórdãos TCU **Projetos e Experimentos**	2	11	6d
Voltamos ao ar! **Meta** downtime, lets-encrypt	0	14	7d
Precisamos realmente de tantos Cientistas de Dados? **Artigos e Publicações** airflow	0	32	18d
Surfando na crista da onda de visualização de dados (Superset e Metabase) **Artigos e Publicações** visualização, dashboard	7	65	19d
On the Philosophical Foundations of Conceptual Models **Artigos e Publicações** modelagem-conceitual	1	19	21d

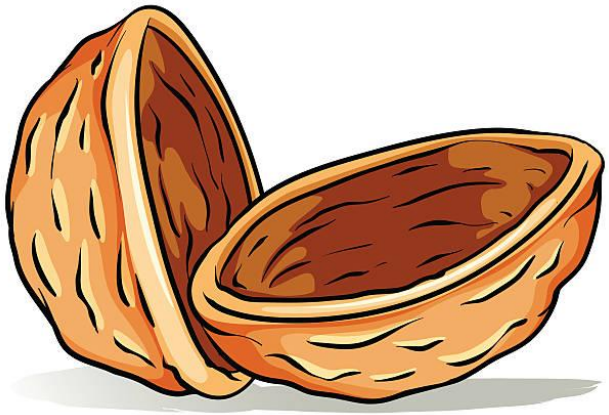
13.
Feedback

Feedback

https://bit.ly/2meiZvs

Com o cadastro do e-mail mandamos esses slides 😉

Inté!



Nitai Bezerra

nitai.silva
@planejamento.gov.br

t. 2020-1551

@nitais

Vitor Bellini

vitor.bellini
@planejamento.gov.br

t. 2020-1504

@podefalar