

# Mentoria Engenharia de Dados



## RESUMO PROFISSIONAL

**Engenheiro de Dados** com +5 anos experiência profissional

**Certificação Astronomer** de Airflow

## FORMAÇÃO

UNIVESP  
Universidade Virtual  
do Estado de São Paulo

**ENGENHARIA**

## EMPRESAS

cargo 

**STARTUP**

 **kovi**

**STARTUP**

Iuri Zambotto



<https://www.linkedin.com/in/iuri-zambotto/>



# Airflow



Apache Airflow é uma plataforma de automação de workflows que permite a definição, monitoramento e gerenciamento de processos complexos. Ela é amplamente utilizada em empresas para automatizar tarefas, como coletar dados, processar informações e executar tarefas críticas.

Automatizar

Automatizar workflows, reduzindo a necessidade de intervenção humana

Consistência

Garantir a consistência e a reproducibilidade dos processos

Desempenho

Oferecer insights sobre o desempenho dos workflows e detectar problemas



# Componentes

Apache Airflow é uma plataforma de automação de workflows que consiste em vários componentes trabalhando juntos para executar e gerenciar DAGs (Directed Acyclic Graphs). A seguir estão os principais componentes do Airflow:

1. **Web Server (airflow webserver)**: O Web Server é o componente responsável por apresentar a interface gráfica do Airflow, permitindo que usuários acessem e gerenciem suas DAGs. Ele também fornece uma API RESTful para que os workflows possam ser controlados programaticamente.
2. **Scheduler (airflow scheduler)**: O Scheduler é o componente responsável por executar as DAGs no Airflow. Ele verifica periodicamente as DAGs para ver se elas estão prontas para serem executadas e, em seguida, inicia a execução delas. O Scheduler também gerencia a fila de tarefas (job queue) e controla o número de trabalhadores (workers) disponíveis.
3. **Worker (airflow worker)**: O Worker é o componente responsável por executar as tarefas (tasks) individuais dentro de uma DAG. Ele comunica com o Scheduler para obter as próximas tarefas a serem executadas e, em seguida, as executa.



# Componentes

Apache Airflow é uma plataforma de automação de workflows que consiste em vários componentes trabalhando juntos para executar e gerenciar DAGs (Directed Acyclic Graphs). A seguir estão os principais componentes do Airflow:

4. **Celery (airflow celery):** Celery é um framework de trabalhadores (worker queue) que permite que os workflows sejam executados de forma assíncron e escalonável. O Airflow utiliza o Celery para gerenciar a fila de tarefas (job queue) e controlar o número de trabalhadores disponíveis.

5. **Flower (airflow flower):** Flower é uma interface gráfica que permite que usuários monitorem e gerenciem os trabalhadores do Airflow. Ele fornece informações sobre as tarefas em andamento, como estado, tempo de execução e erros, o que ajuda a monitorar e depurar os workflows.





# Executores do Apache Airflow

Os Executores são os responsáveis por executar as tarefas (tasks) individuais dentro de uma DAG no Airflow. Existem vários tipos de Executores no Airflow, cada um com suas características e vantagens. A seguir estão os principais Executores do Airflow.

1. **LocalExecutor (airflow.local\_executor):** O LocalExecutor é o executor padrão do Airflow. Ele executa as tarefas localmente no mesmo servidor onde o Web Server está rodando.
2. **CeleryExecutor (airflow.celery\_executor):** O CeleryExecutor utiliza o framework de trabalhadores (worker queue) Celery para executar as tarefas em um pool de servidores remotos. Isso permite que os workflows sejam escalonados e executados em vários servidores, aumentando a capacidade de processamento e reduzindo a carga no servidor principal.
3. **SequentialExecutor (airflow.sequential\_executor):** O SequentialExecutor é um executor sequencial que executa as tarefas uma após a outra, sem paralelizar nenhuma delas. Isso pode ser útil quando as tarefas têm dependências ou interações específicas.
4. **KubernetesExecutor (airflow.kubernetes\_executor):** O KubernetesExecutor é um executor que utiliza o container orchestrator Kubernetes para executar as tarefas em containers Docker. Isso permite que os workflows sejam executados em um ambiente controlado e escalonável, com recursos de vários servidores.



# Principais conceitos

Airflow é uma ferramenta de automação de workflows e processos que fornece um ambiente robusto para gerenciar e executar tarefas em diferentes plataformas. Aqui estão os principais conceitos do Airflow:

- 1) **Operators**
- 2) **Hooks**
- 3) **Sensors**
- 4) **Connections**





# 1. Operators (Operadores)

Os Operadores são a base da funcionalidade do Airflow. Eles são responsáveis por executar as tarefas específicas no workflow, como enviar emails, realizar upload de arquivos para um bucket S3 ou executar scripts Python.

## *Exemplos de Operadores:*

- **BashOperator:** Executa comandos Bash em um servidor.
- **PythonOperator:** Executa scripts Python.
- **EmailOperator:** Envia emails.



# 1. Operators (Operadores)

The screenshot displays the Apache Airflow web interface. At the top, the navigation menu includes: Airflow, DAGs, Cluster Activity, Datasets, Security, Browse, Admin, and Docs. The current time is 17:02 UTC. Below the navigation menu, there are tabs for Grid, Graph (selected), Calendar, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, and Audit Log. A search bar for DAG Docs is present. The main content area shows a DAG run for 'tutorial' on 2023-09-19 at 16:28:42 UTC. The run is in a 'success' state. The DAG graph shows three tasks: 'print\_date' (success), 'sleep' (success), and 'templated' (failed). The 'templated' task is highlighted in red, indicating a failure. The interface also includes a filter bar with options like 'All Run Types' and 'All Run States', and a 'Clear Filters' button. A legend at the bottom right shows the status of the tasks: 'print\_date' (success), 'sleep' (success), and 'templated' (failed).



## 2. Hooks (Ferramentas)

As Ferramentas são uma extensão dos Operadores que fornece funcionalidades adicionais para interagir com diferentes plataformas e serviços, como AWS, GCP, Azure, etc

### *Exemplos de Ferramentas:*

- **AWSHook:** Fornece acesso a recursos AWS, como EC2, S3, SQS, etc.
- **GCPHook:** Fornece acesso a recursos GCP, como Cloud Storage, Cloud SQL, etc.
- **EmailHook:** Envia emails utilizando serviços como Sendgrid ou Mailgun.



# 3. Sensors (Sensores)

Os Sensores são utilizados para monitorar o estado de uma tarefa e aguardar que ela seja concluída antes de continuar com o workflow.

## Exemplos de Sensores:

- **FileSensor:** Verifica se um arquivo foi criado ou modificado.
- **SSH Sensor:** Verifica o estado de uma máquina remota utilizando SSH.
- **HTTPSensor:** Verifica se um recurso HTTP está disponível.



# 4. Connections (Conexões)

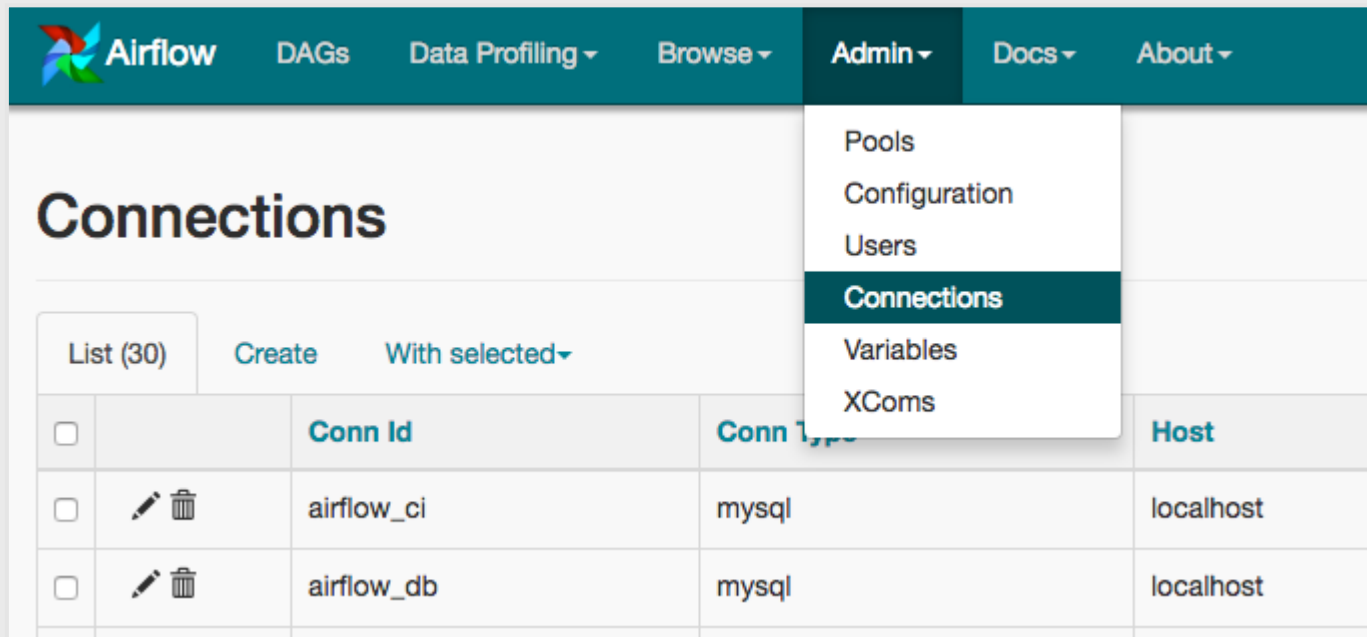
As Conexões são configurações que permitem ao Airflow se conectar a diferentes plataformas e serviços, como AWS, GCP, Azure, etc.

## Exemplos de Conexões:

- **AWS Connection:** Configuração para se conectar à conta AWS.
- **GCP Connection:** Configuração para se conectar à conta GCP.
- **Email Connection:** Configuração para se conectar a um serviço de email como Sendgrid ou Mailgun.







# 4. Connections (Conexões)



The screenshot displays the Airflow web interface. At the top, there is a navigation bar with the Airflow logo and several menu items: DAGs, Data Profiling, Browse, Admin, Docs, and About. The 'Admin' menu is currently open, showing a list of options: Pools, Configuration, Users, Connections (which is highlighted), Variables, and XComs.

Below the navigation bar, the main heading is 'Connections'. Underneath this heading, there are several controls: a 'List (30)' button, a 'Create' button, and a 'With selected' dropdown menu.

The main content area features a table with the following columns: a checkbox, an edit/delete icon, 'Conn Id', 'Conn Type', and 'Host'. The table contains two rows of data:

<input type="checkbox"/>		Conn Id	Conn Type	Host
<input type="checkbox"/>	 	airflow_ci	mysql	localhost
<input type="checkbox"/>	 	airflow_db	mysql	localhost



# 5. Variables (Variáveis)

No Apache Airflow, uma Variável é um par chave-valor que pode ser utilizado para armazenar e recuperar valores ao longo de diferentes tarefas, DAGs e até mesmo todo o workflow. As variáveis são uma característica poderosa que permite dissociar a configuração do seu fluxo de trabalho da sua lógica. Ao usar variáveis de forma eficaz, você pode tornar seus fluxos de trabalho no Airflow mais flexíveis, reutilizáveis e fáceis de manter!

## Tipos de Variáveis

O Airflow fornece três tipos de variáveis:

- 1. Variáveis de Ambiente:** Essas são variáveis definidas no nível de ambiente, o que significa que podem ser acessadas por qualquer tarefa dentro desse ambiente.
- 2. Variáveis de DAG:** Essas são variáveis específicas para um DAG e apenas podem ser acessadas por tarefas dentro desse DAG.
- 3. Variáveis de Tarefa:** Essas são variáveis específicas para uma única tarefa e apenas podem ser acessadas pelaquela tarefa.



# 5. Variables (Variáveis)

No Apache Airflow, uma Variável é um par chave-valor que pode ser utilizado para armazenar e recuperar valores ao longo de diferentes tarefas, DAGs e até mesmo todo o workflow. As variáveis são uma característica poderosa que permite dissociar a configuração do seu fluxo de trabalho da sua lógica. Ao usar variáveis de forma eficaz, você pode tornar seus fluxos de trabalho no Airflow mais flexíveis, reutilizáveis e fáceis de manter!

## Usando Variáveis no Airflow

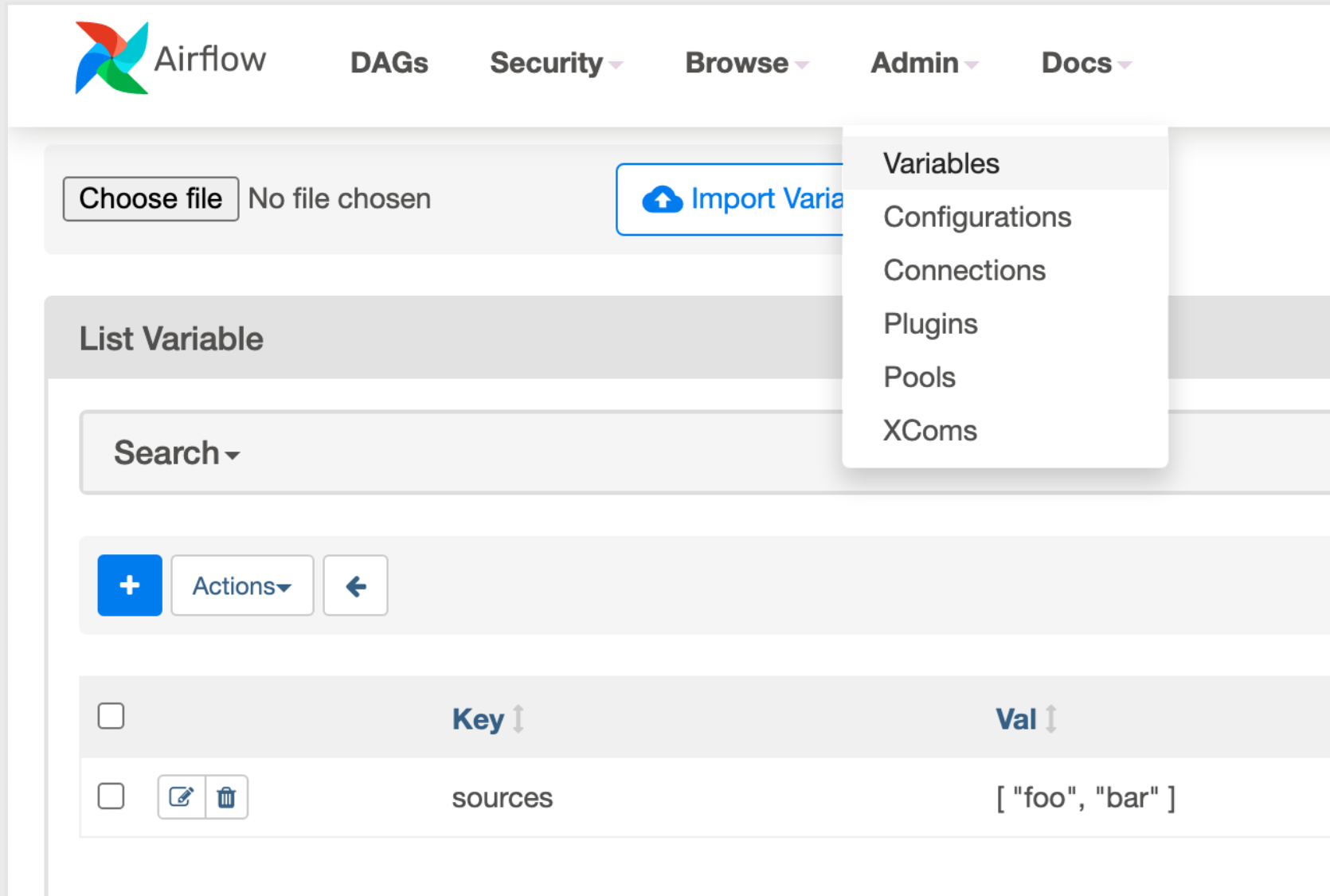
Você pode usar variáveis em várias maneiras:

- 1. Passando variáveis como argumentos:** Você pode passar variáveis como argumentos para operadores, como ``bash_operator`` ou ``python_operator``.
- 2. Acessando variáveis dentro de tarefas:** Você pode acessar variáveis dentro de uma tarefa usando a sintaxe ``{{ var.value }}``.
- 3. Usando variáveis em modelos:** Você pode usar variáveis em arquivos de modelo (por exemplo, templates Jinja) para gerar conteúdo dinâmico.





# 5. Variables (Variáveis)



The screenshot shows the Airflow web interface. At the top, there is a navigation bar with the Airflow logo and several menu items: DAGs, Security, Browse, Admin, and Docs. The 'Admin' menu is open, showing a list of options: Variables, Configurations, Connections, Plugins, Pools, and XComs. The 'Variables' option is highlighted. Below the navigation bar, there is a file upload section with a 'Choose file' button and 'No file chosen' text, and an 'Import Variables' button. Below that is a 'List Variable' section with a search bar and a table of variables. The table has columns for a checkbox, 'Key', and 'Val'. One variable is listed with the key 'sources' and the value ['foo', 'bar'].

Airflow

DAGs Security Browse Admin Docs

Choose file No file chosen Import Variables

List Variable

Search

+ Actions

<input type="checkbox"/>	Key ↑	Val ↓
<input type="checkbox"/>	sources	[ "foo", "bar" ]



# 6. XComs (Cross-Task Communication)

Os XComs são uma forma de comunicação entre tarefas no Airflow, permitindo que as tarefas compartilhem dados e resultados entre si.

## Exemplos de XComs:

- Passar um valor de uma tarefa para outra.
- Compartilhar um arquivo gerado por uma tarefa com outra tarefa.
- Notificar a outro workflow sobre o resultado de uma tarefa.

















# 6. XComs (Cross-Task Communication)

Airflow DAGs Data Profiling Browse Admin Docs About 2019-11-07 05:18:27 UTC

## Xcoms

List (231) Create Add Filter With selected Search: key, timestamp

<input type="checkbox"/>		Key	Value	Timestamp	Execution Date	Task Id	Dag Id
<input type="checkbox"/>	 	test_dag	{'key1': 'value1'}	2019-11-06 07:04:42.534514+00:00	2019-11-05 00:48:00+00:00	xcom_push_task	write_to_xcom
<input type="checkbox"/>	 	test_dag	{'key1': 'value1'}	2019-11-06 07:04:47.838219+00:00	2019-11-05 00:51:00+00:00	xcom_push_task	write_to_xcom
<input type="checkbox"/>	 	test_dag	{'key1': 'value1'}	2019-11-06 07:04:51.009360+00:00	2019-11-05 00:39:00+00:00	xcom_push_task	write_to_xcom
<input type="checkbox"/>	 	test_dag	{'key1': 'value1'}	2019-11-06 07:04:56.234171+00:00	2019-11-05 00:54:00+00:00	xcom_push_task	write_to_xcom
<input type="checkbox"/>	 	test_dag	{'key1': 'value1'}	2019-11-06 07:05:07.818095+00:00	2019-11-05 01:00:00+00:00	xcom_push_task	write_to_xcom
<input type="checkbox"/>	 	test_dag	{'key1': 'value1'}	2019-11-06 07:05:21.093908+00:00	2019-11-05 01:03:00+00:00	xcom_push_task	write_to_xcom
<input type="checkbox"/>	 	test_dag	{'key1': 'value1'}	2019-11-06 07:05:28.601755+00:00	2019-11-05 01:06:00+00:00	xcom_push_task	write_to_xcom

