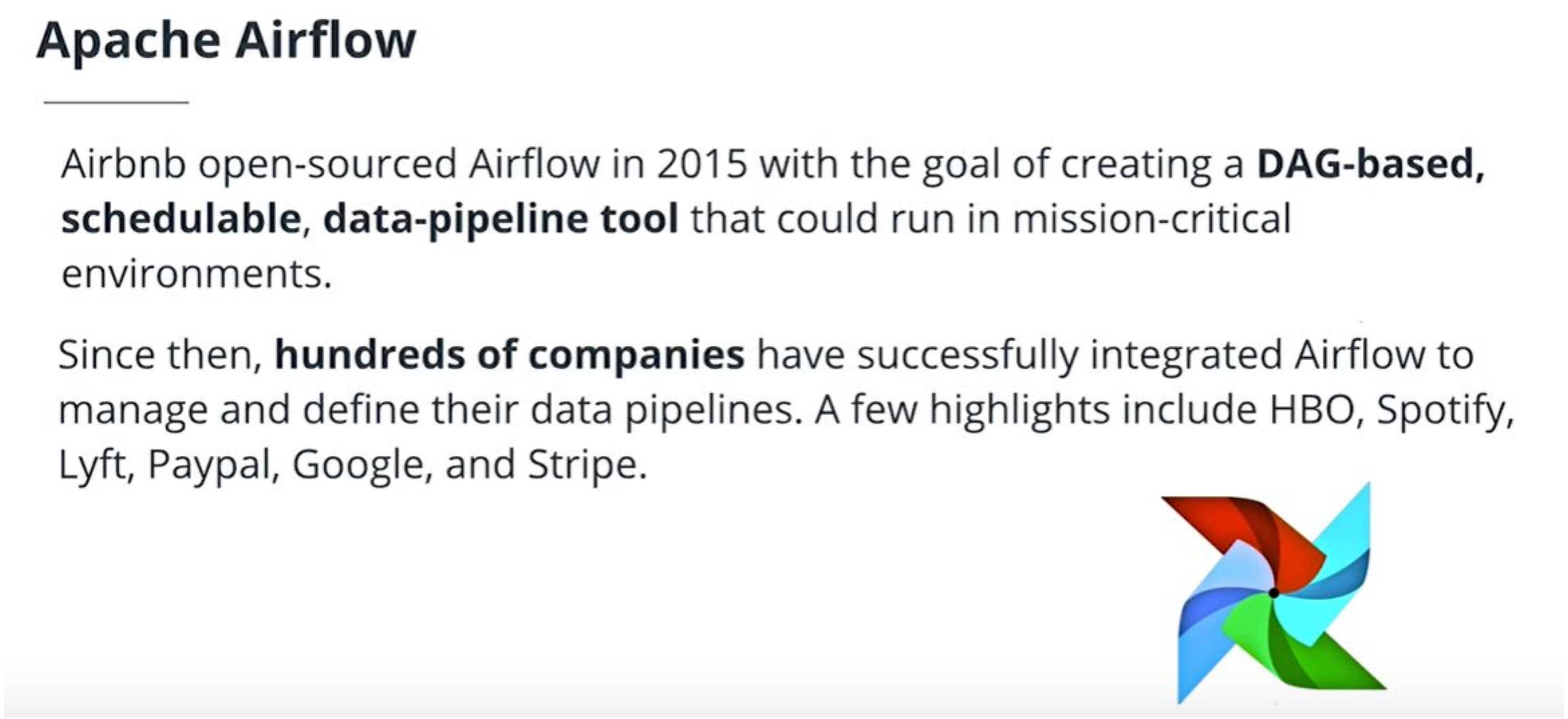
Data Pipelines with Airflow

What is a data pipeline?

What is a data pipeline?

- A series of steps in which data is processed.
- Usually can be on batch or streaming
- Extract Transform Load (ETL) and Extract Load Transform (ELT) are common patterns found in data pipelines

Apache Airflow

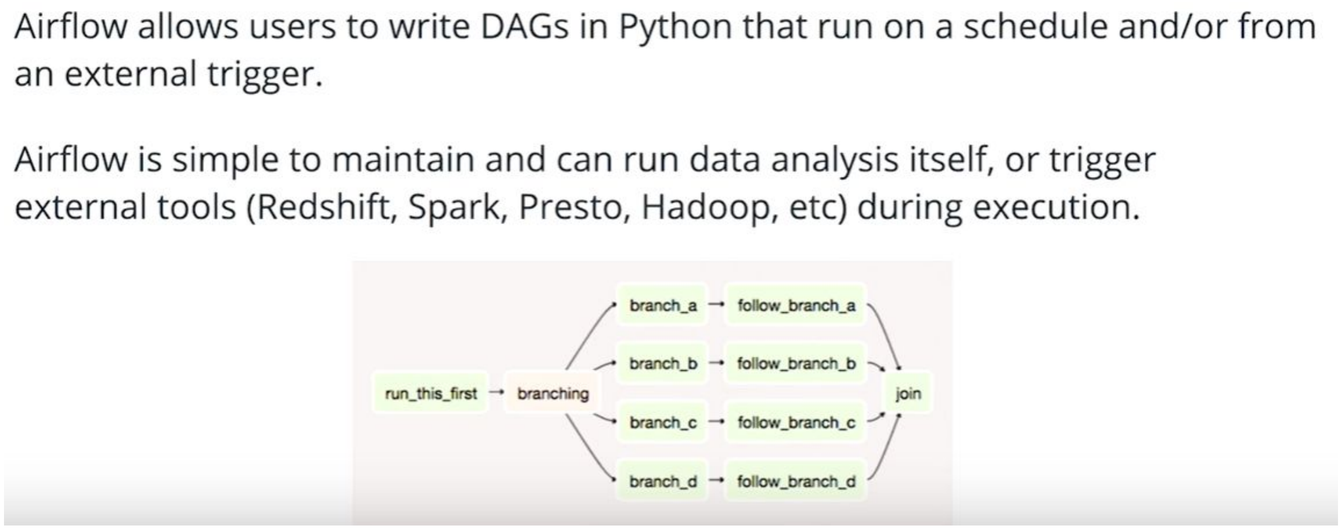


Apache Airflow

Airbnb open-sourced Airflow in 2015 with the goal of creating a **DAG-based, schedulable**, **data-pipeline tool** that could run in mission-critical environments.

Since then, **hundreds of companies** have successfully integrated Airflow to manage and define their data pipelines. A few highlights include HBO, Spotify, Lyft, Paypal, Google, and Stripe.

It can help you to orchestrate complex computational workflows and data processing pipelines. If you find yourself running cron task which execute ever longer scripts, or keeping a calendar of big data processing batch jobs then Airflow can probably help you.



Airflow allows users to write DAGs in Python that run on a schedule and/or from an external trigger.

Airflow is simple to maintain and can run data analysis itself, or trigger external tools (Redshift, Spark, Presto, Hadoop, etc) during execution.

What is a DAG ?

- Data Pipelines are well expressed as Directed Acyclic Graphs.
- Directed Acyclic Graphs (DAGs): DAGs are a special subset of graphs in which the edges between nodes have a specific direction, and no cycles exist. When we say “no cycles exist” what we mean is the nodes cant create a path back to themselves.
- Nodes**:** A step in the data pipeline process.
- Edges: The dependencies or relationships other between nodes.

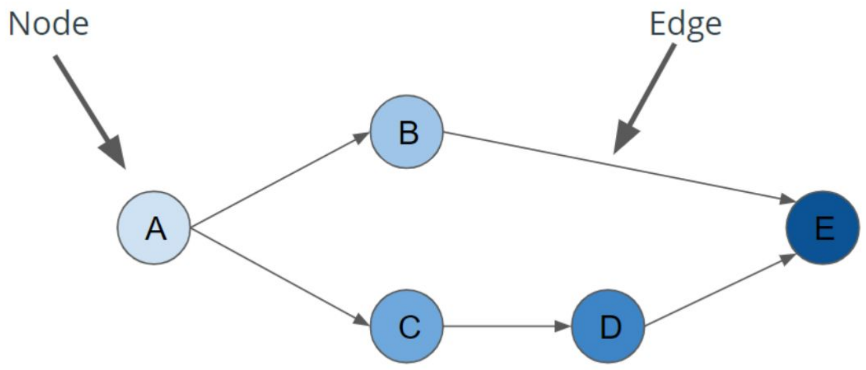
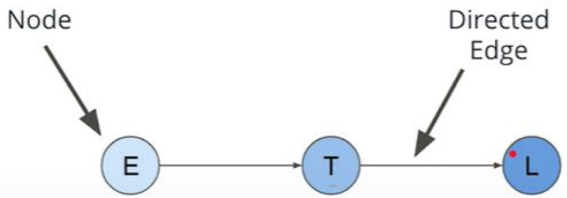


Diagram of a Directed Acyclic Graph

Data Pipelines as DAGs

In ETL, each step of the process typically depends on the last.

Each step is a *node* and the dependencies on prior steps are *directed edges.*



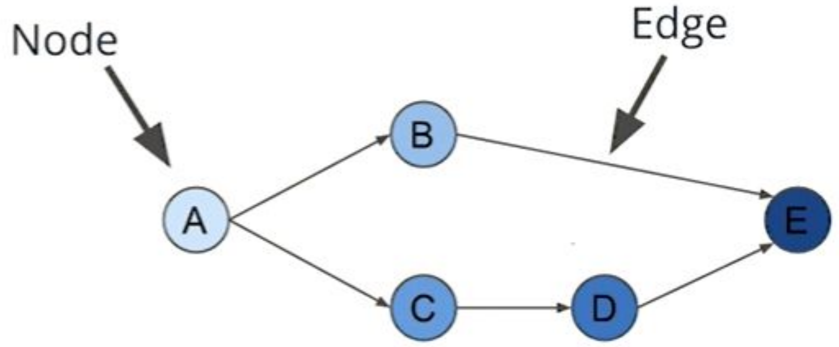
Creating a DAG

Demo1

- Airflow DAGs are composed of Tasks.
- Each Task is created by instantiating an Operator class.

In Airflow DAGs:

- Nodes = Tasks
- Edges = Ordering and dependencies between tasks



exercise1

Schedules

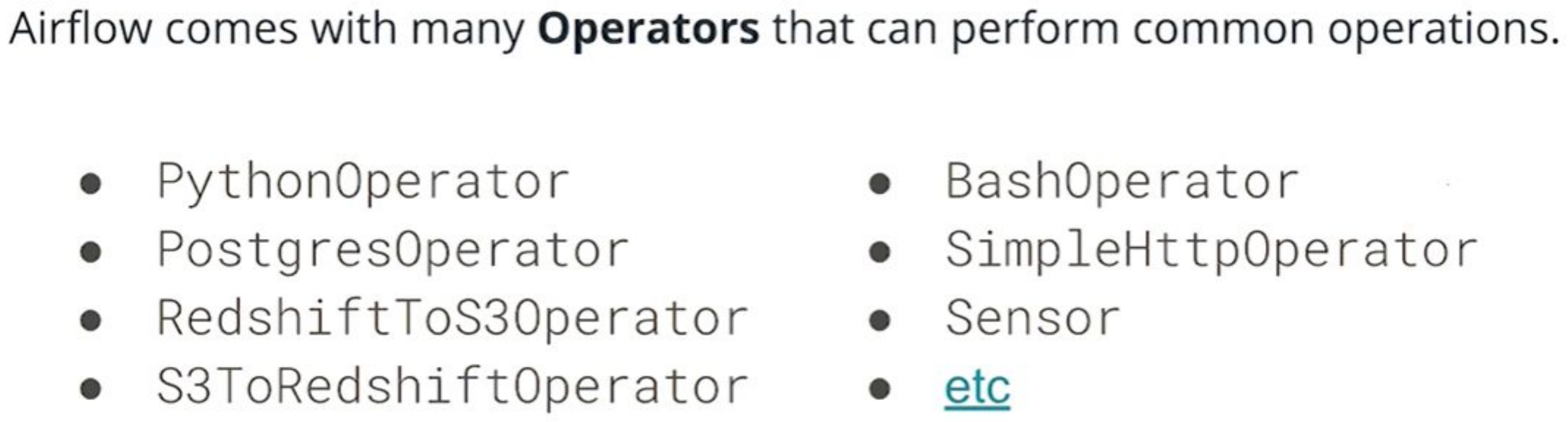
Schedules are optional, and may be defined with cron strings or Airflow Presets.

Airflow provides the following presets

- `'@once` - Run a DAG once and then never again
- `'@hourly'` - Run the DAG every hour
- `'@daily'` - Run the DAG every day
- `'@weekly'` - Run the DAG every week
- `'@monthly'` - Run the DAG every month
- `'@yearly'` - Run the DAG every year
- `None` - Only run the DAG when the user initiates it

Demo 2 - exercise 2

Operators and Tasks



Airflow comes with many **Operators** that can perform common operations.

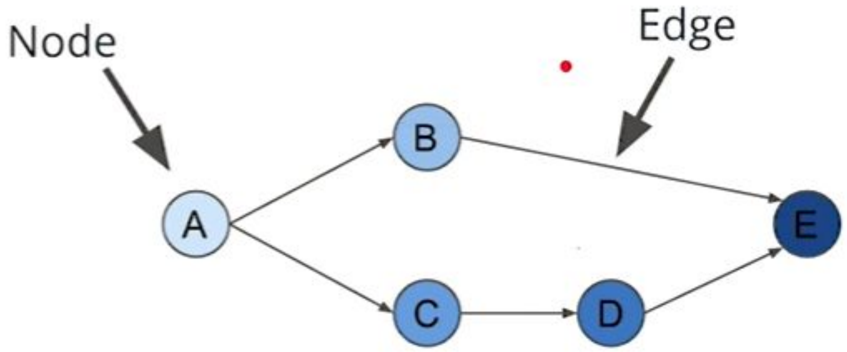
- `PythonOperator`
- `PostgresOperator`
- `RedshiftToS3Operator`
- `S3ToRedshiftOperator`
- `BashOperator`
- `SimpleHttpOperator`
- `Sensor`
- etc

Demo_python_operator - demo_simplehttp_operator

Task Dependencies

In Airflow DAGs:

- Nodes = Tasks
- Edges = Ordering and dependencies between tasks



Task dependencies can be described programmatically in Airflow using >> and <<

- a >> b means a comes before b
- a << b means a comes after b

Demo 3 - `my_awesome_datapipeline_exercise`

Create my own operator

- Demo_my_operator
- Get_data_operator
- my_awesome_datapipeline_exercise-2