Operator Relationships and Bitshift Composition

Organize your tasks

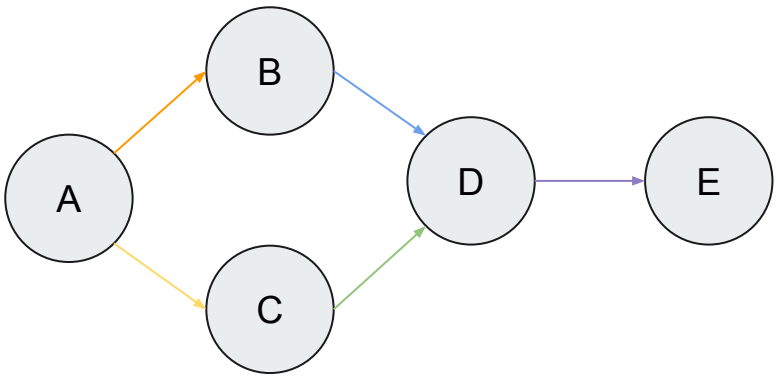
Quick Reminder

If you remember from previous lessons we have seen that a DAG describes a collection of tasks organized in a way that reflects their relationships and dependencies.

A DAG is nothing more than a directed acyclic (no loops) graph with nodes (tasks) connected by edges (dependencies)

Schema

- In this DAG, nodes are A, B, C, D and E
- Edges are the colored arrows.
- Each arrow represents a dependency
  - B depends of A
  - C depends of A
  - D depends of B and C
  - E depends of D

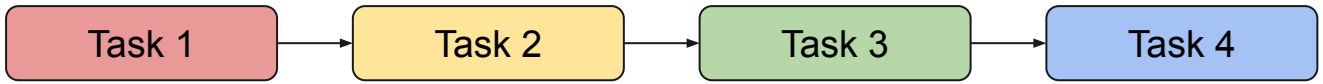


How to Make Dependencies in Airflow?

- There are two ways of describing dependencies between operators in Apache Airflow:
  - By using the traditional operator relationships with
    - `set_upstream()`
    - `set_downstream()`
  - From Apache Airflow 1.8 you can use Python bitshift operators
    - `<< ( = set_upstream )`
    - `>> ( = set_downtream )`

Example

The dependencies of the DAG below can be described in 4 ways:



- `t1.set_downstream(t2); t2.set_downstream(t3); t3.set_downstream(t4)`
- `t4.set_upstream(t3); t3.set_upstream(t2); t2.set_upstream(t1)`
- `t1 >> t2 >> t3 >> t4`
- `t4 << t3 << t2 << t1`

Let’s do it!

Let’s see how to code this into our DAG!