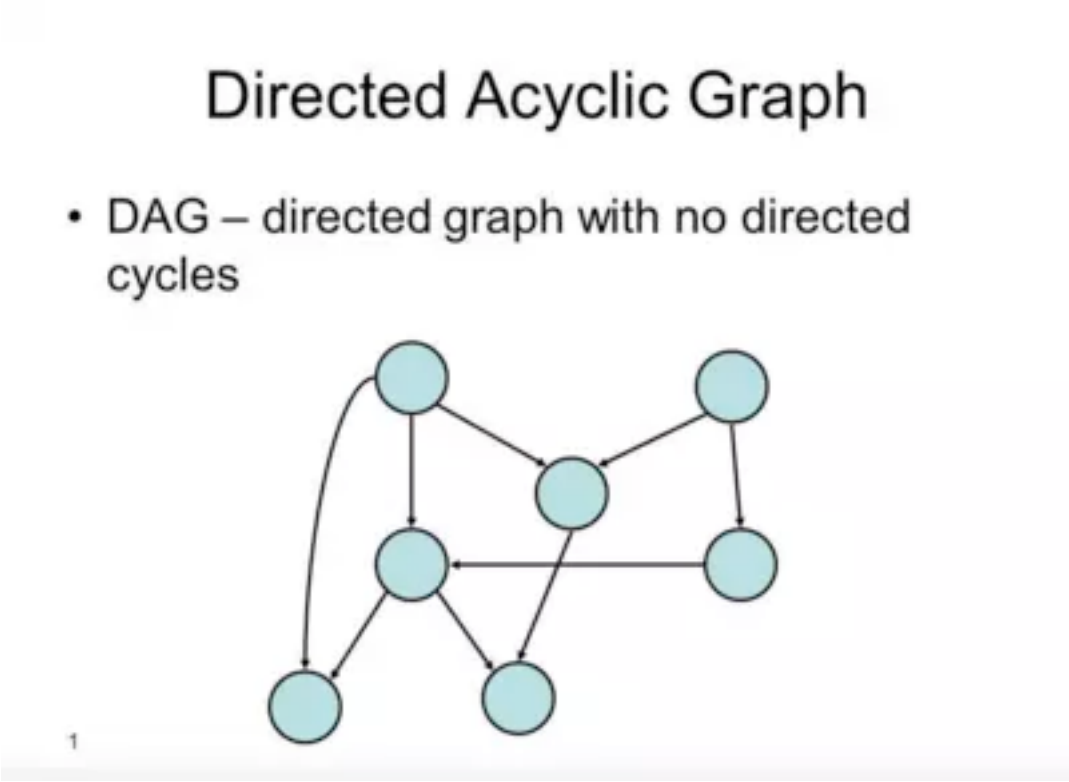
Airflow là gì?

- Airflow là một công cụ mã nguồn mở miễn phí được sử dụng bởi Apache để quản lý quy trình làm việc
- Đây là một trong những hệ thống quản lý quy trình làm việc phổ biến nhất và tốt nhất hiện có, với sự hỗ trợ mạnh mẽ của cộng đồng.

DAG là gì?

- DAG (Directed Acyclic Graph) có nghĩa là biểu đồ xoay chiều có hướng
- Có hướng nghĩa luồng là một chiều
- Acyclic có nghĩa là luồng không bao giờ quay trở lại nút đầu tiên (đỉnh) / nút bắt đầu (ngăn chặn luồng hai chiều), điều này hơi giống với giao tiếp simplex.
- Biểu đồ là một cấu trúc dữ liệu phi tuyến tính với hai thành phần
    - Nút (đỉnh). Các nút đại diện cho các đối tượng được kết nối với nhau, thường sử dụng dấu chấm đậm
    - Cạnh: Một cạnh kết nối nút này với nút khác, nếu không có mũi tên tiến thì có nghĩa là luồng có hai chiều, ngược lại nếu có mũi tên thì có nghĩa là luồng chỉ có một hướng.
- Nút được chỉ đến bởi mũi tên được cho là có sự phụ thuộc vào nút được chỉ bởi mũi tên.



Operator

Operator là gì và tại sao chúng ta cần chúng?

- Trong thực tế, DAGs chỉ đại diện cho quy trình công việc và không thực hiện bất kỳ tính toán nào (hoặc thực hiện các hoạt động). Vì vậy, để thực hiện hoạt động thực tế, chúng ta cần các Operators. Trong trường hợp này, operator là placeholders giúp chúng ta xác định những gì chúng ta muốn làm.
- Có nhiều loại operator khác nhau trong Airflơ tùy thuộc vào yêu cầu của bạn và ngôn ngữ lập trình bạn đang sử dụng. BashOperator, PythonOperator được sử dụng phổ biến nhất. Có thể truy cập (thư mục operator chính thức của Airflow trên Github): liên kết github repo để xem tất cả các loại operator. Ngay cả khi operator bạn cần không có ở đó, rất có thể bạn sẽ sớm thấy nó trong Airflow / Contrib (github directory: liên kết thư mục đóng góp) nhờ sự hỗ trợ tuyệt vời của cộng đồng và tất cả các nhà phát triển.
- *BaseOperator* là lớp cha của tất cả các toán tử, tức là tất cả các toán tử sẽ kế thừa các thuộc tính của lớp *BaseOperator* .
- Ví dụ. Giả sử bạn muốn chạy một hàm Python, thì để thực thi nó, bạn cần sử dụng PythonOperator.

Phân loại các operator.

- **Cảm biến.** Ví dụ: HdfsSensor (cảm biến hệ thống tệp Hadoop sẽ chạy cho đến khi phát hiện tệp hoặc thư mục trong HDFS).
- **Operator.** Ví dụ, PythonOperator (sẽ giúp thực thi các hàm được viết bằng Python).
- **Truyền tải. Ví dụ** , MySqlToHiveTransfer (như tên cho thấy, nó sẽ chuyển dữ liệu từ MySQL sang Hive).

```
# sample python function
def print_function(x):
    return x + " is best tool for workflow management"

# defining the DAG
dag = DAG(
    'python_operator_sample',
    default_args=default_args,
    description='Demo on How to use the Python Operator?',
    schedule_interval=timedelta(days=1),
)

#creating the task
t1 = PythonOperator(
    task_id='print',
    python_callable= print_function,
    op_kwargs = {"x" : "Apache Airflow"},
    dag=dag,
)

t1
```

Task .

Task là gì?

- Trong Airflow, việc khởi tạo một operator (được gán cho một tên cụ thể) được gọi là một task. Trong Airflow, mỗi operator có một task, đây là cách viết task tiêu chuẩn. Khi xác định task, bạn phải xác định *task_id* và vùng chứa *dag* . Ví dụ:

```
#creating the task
t1 = PythonOperator(
    task_id='print',
    python_callable= print_function,
    op_kwargs = {"x" : "Apache Airflow"},
    dag=dag,
)
```

Dagruns là gì?

- DagRuns có thể được định nghĩa là các DAG được lên lịch chạy sau một khoảng thời gian nhất định, chẳng hạn như ngày / phút / giờ, v.v.
- Chúng tôi cũng có execution_time bắt đầu tại start_time (bên trong DAG) và tiếp tục thực hiện sau một khoảng thời gian nhất định theo khoảng thời gian được xác định bởi schedule_interval (bên trong DAG).

TaskInstances là gì?

- Các tác vụ được định nghĩa trong Dagruns được gọi là TaskInstances.
- Chúng ta phải có trạng thái liên quan đến TaskInstances và Dagruns, chẳng hạn như "running", "failed", "queued", "skipped", v.v.

Vai trò của task dependencies là gì?

- Sau khi thiết lập DAG và các nhiệm vụ, chúng ta có thể chuyển sang thiết lập dependencies, thứ tự mà chúng ta muốn thực hiện các task (dựa trên DAG mà chúng ta có).
- Giả sử chúng ta muốn task t2 thực hiện sau task t1, thì chúng ta sẽ gửi phụ thuộc của t2 vào t1 (nghĩa là chỉ sau khi t1 kết thúc công việc thì mới có t2 để chạy). Có hai cách để thiết lập các phụ thuộc.
    - Sử dụng set_upstream / set_downstream. Ví dụ: *t1.set_downstream (t2)*
    - Sử dụng toán tử dịch (>> hoặc <<). Ví dụ: *t1 >> t2*

Tổng kết.

- Airflow là một công cụ quản lý quy trình làm việc miễn phí mã nguồn mở của Apache, có lẽ là công cụ tốt nhất hiện có.
- *Tìm hiểu thêm về công cụ này? Truy cập tài liệu chính thức*